Autonomous and Mobile Robotics

Prof. Giuseppe Oriolo

MPC-Based Humanoid Gait Generation with application to Pursuit-Evasion

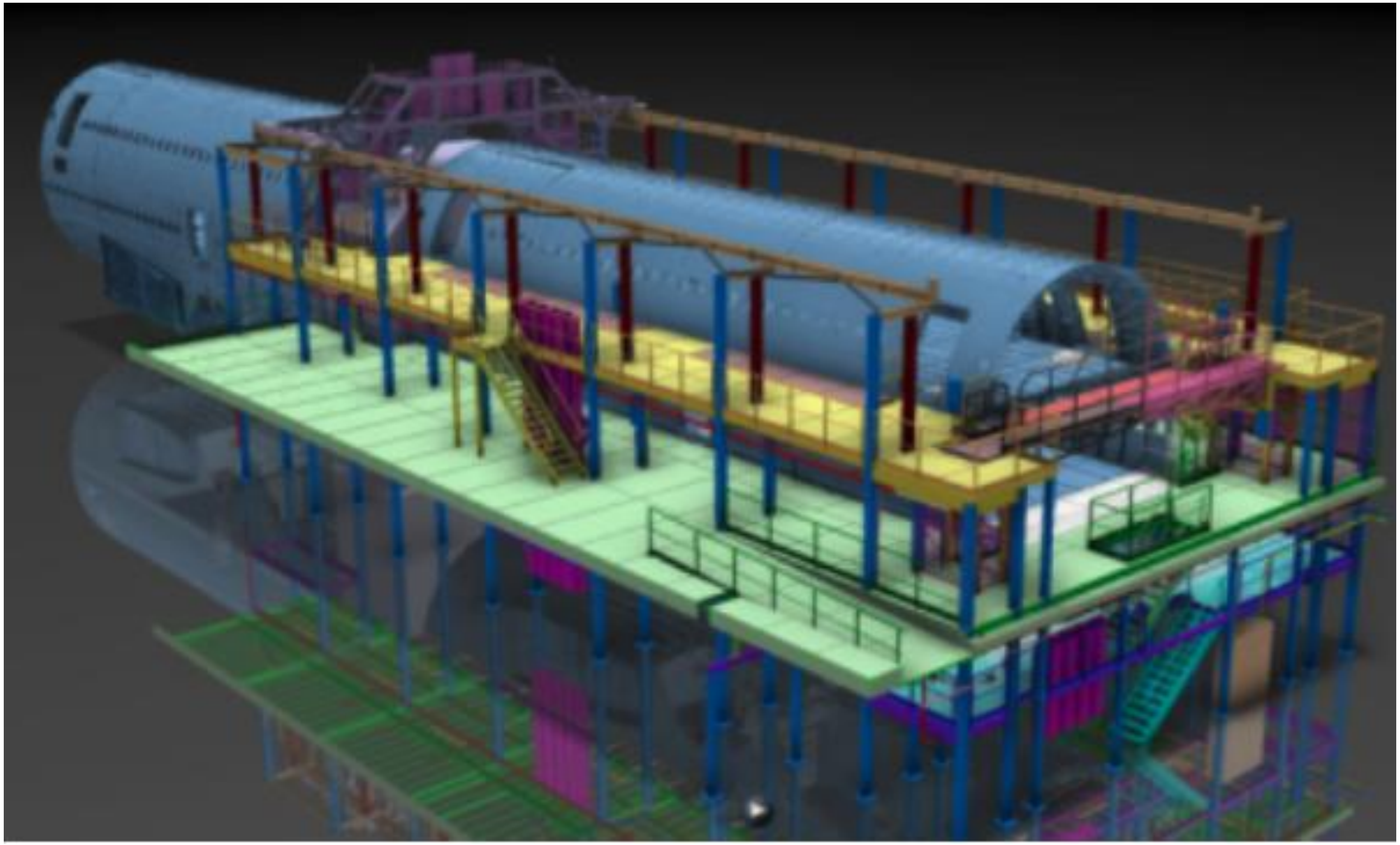
(slides prepared by Nicola Scianca and Daniele De Simone)

Dipartimento di Ingegneria Informatica
Automatica e Gestionale Antonio Ruberti

Sapienza
Università di Roma

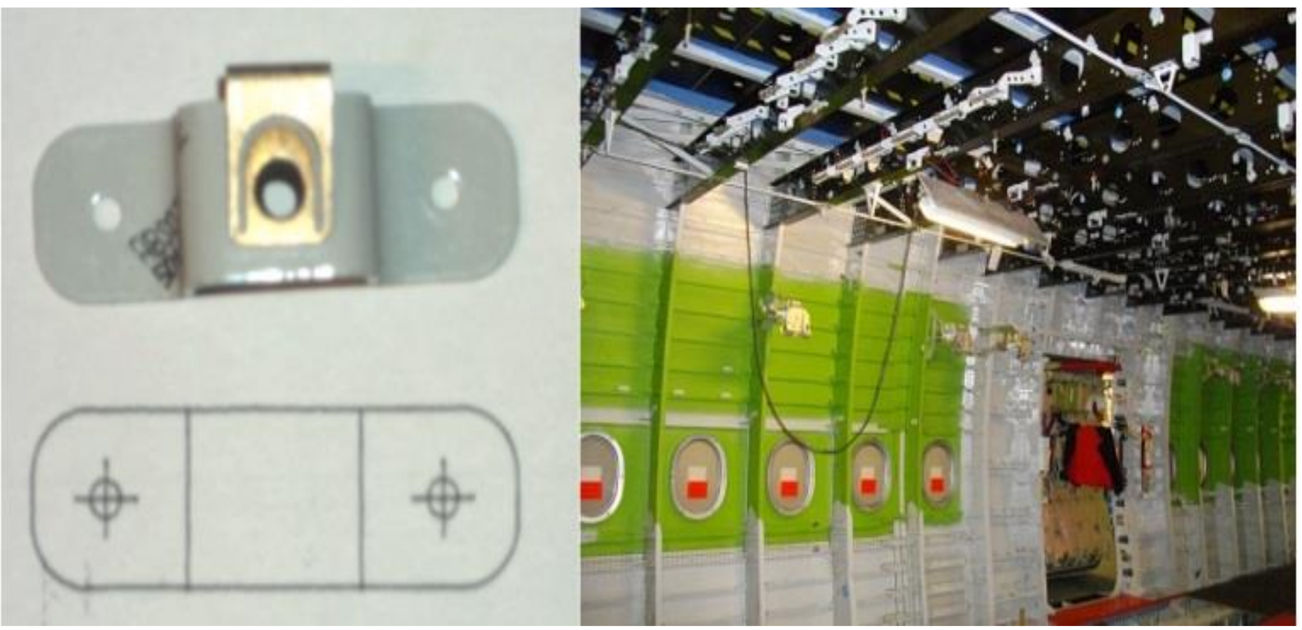
Introduction

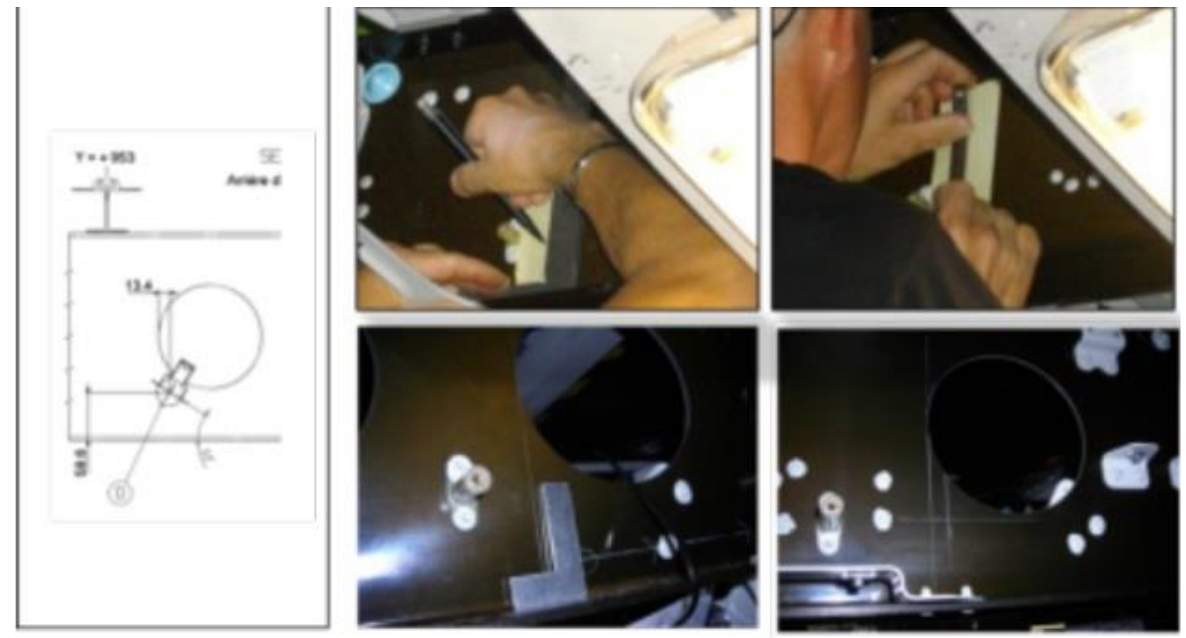
- COMANOID: Multi-contact Collaborative Humanoids in Aircraft Manufacturing started on January 1, 2015



Introduction

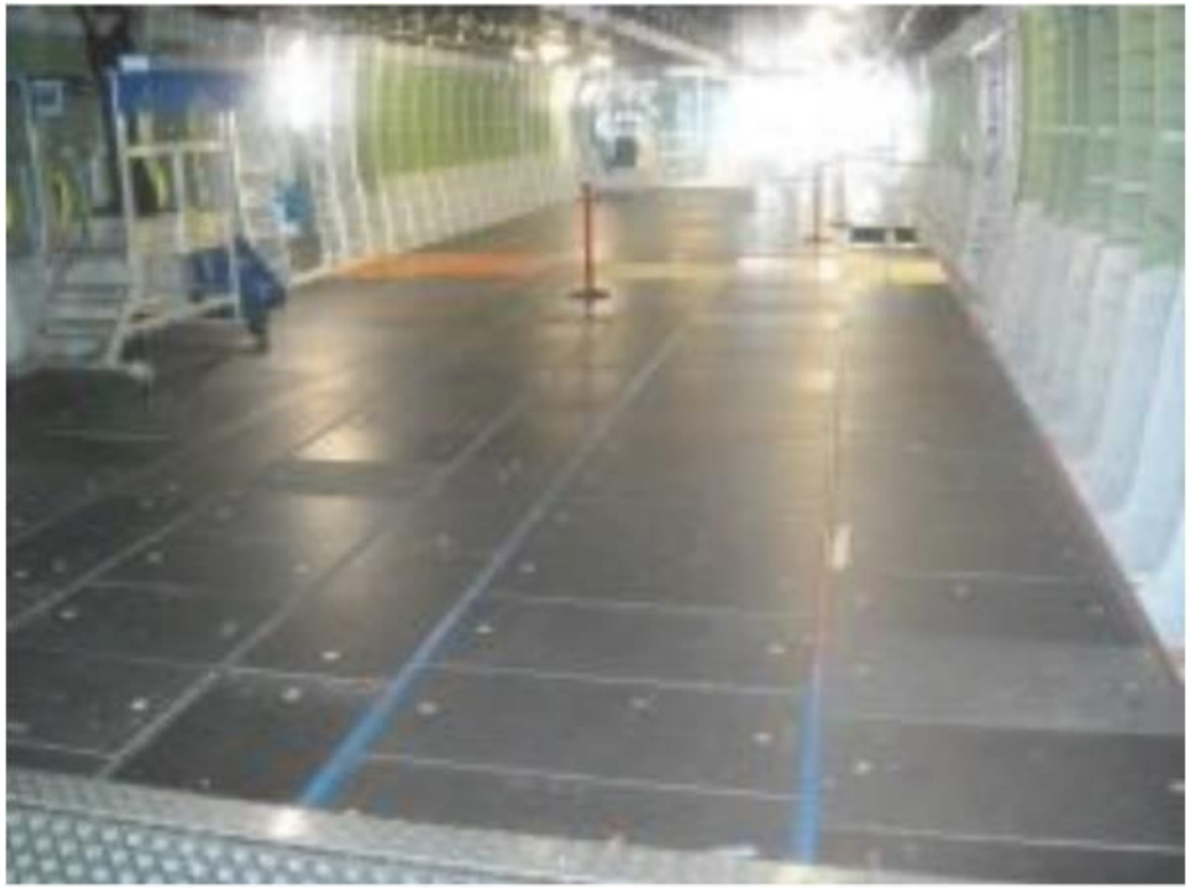
- Automate the process of printing brackets for wires in the aircraft





Introduction

- Airbus Group ready to deploy COBOT in production
- COBOT can work only in the 60% of the environment

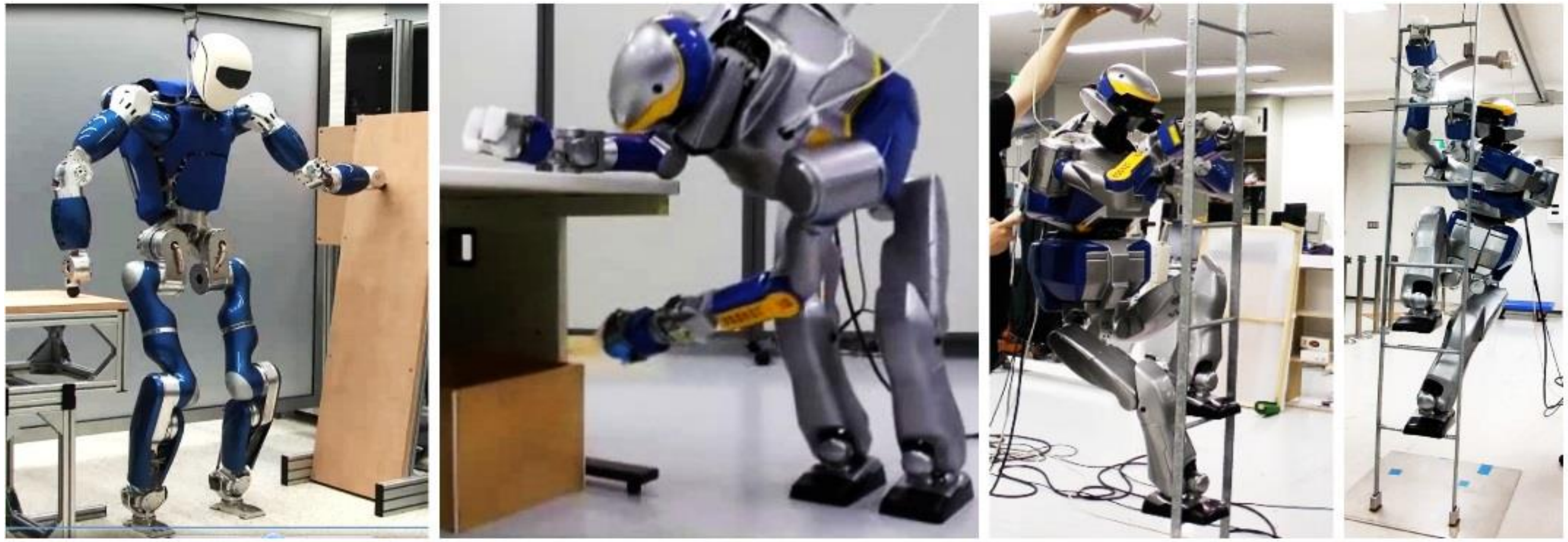


Introduction

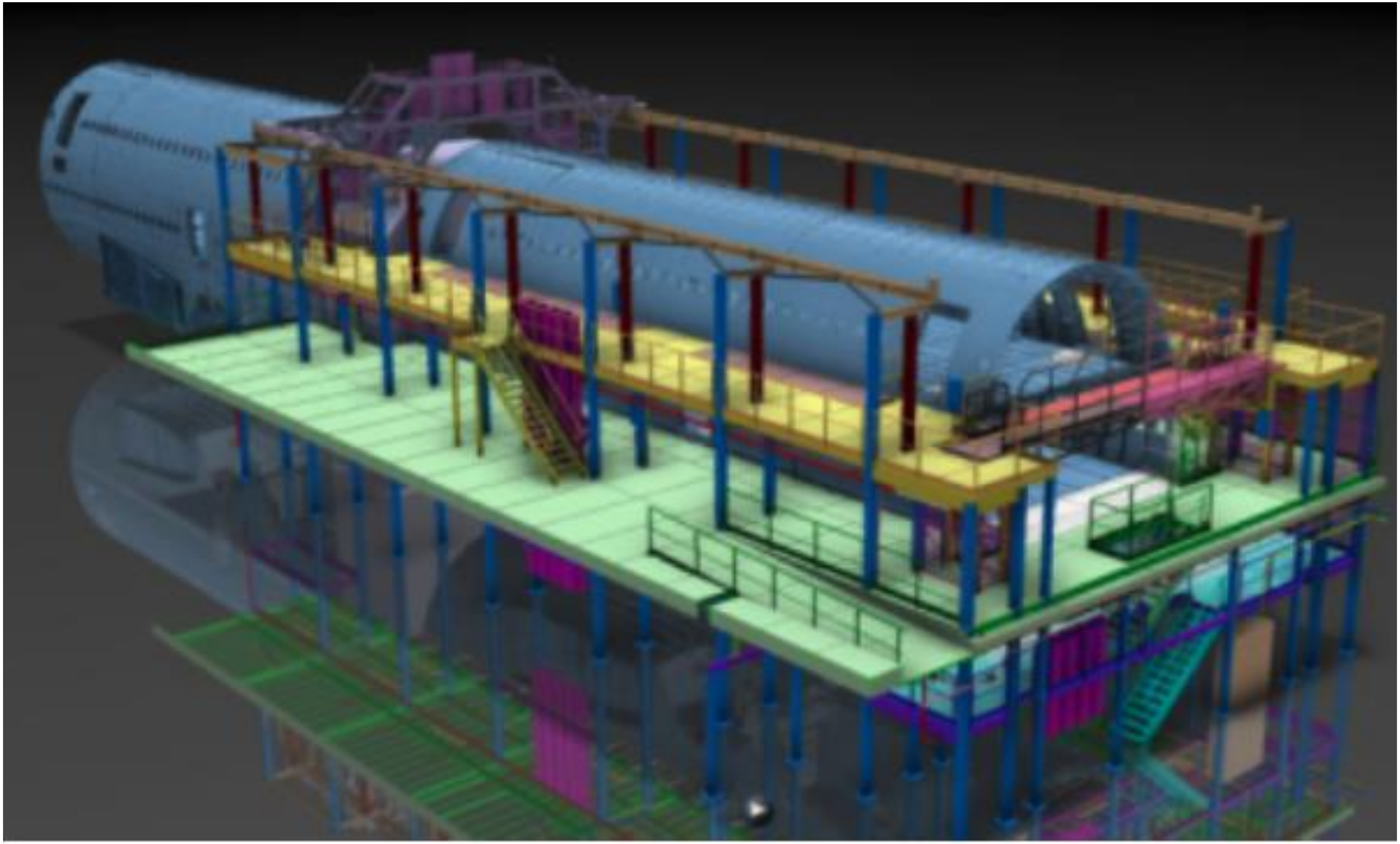
- Environment too complex for a wheeled robot

Introduction

- Tasks are achieved in constrained or hard postures
- Multi-contact situations

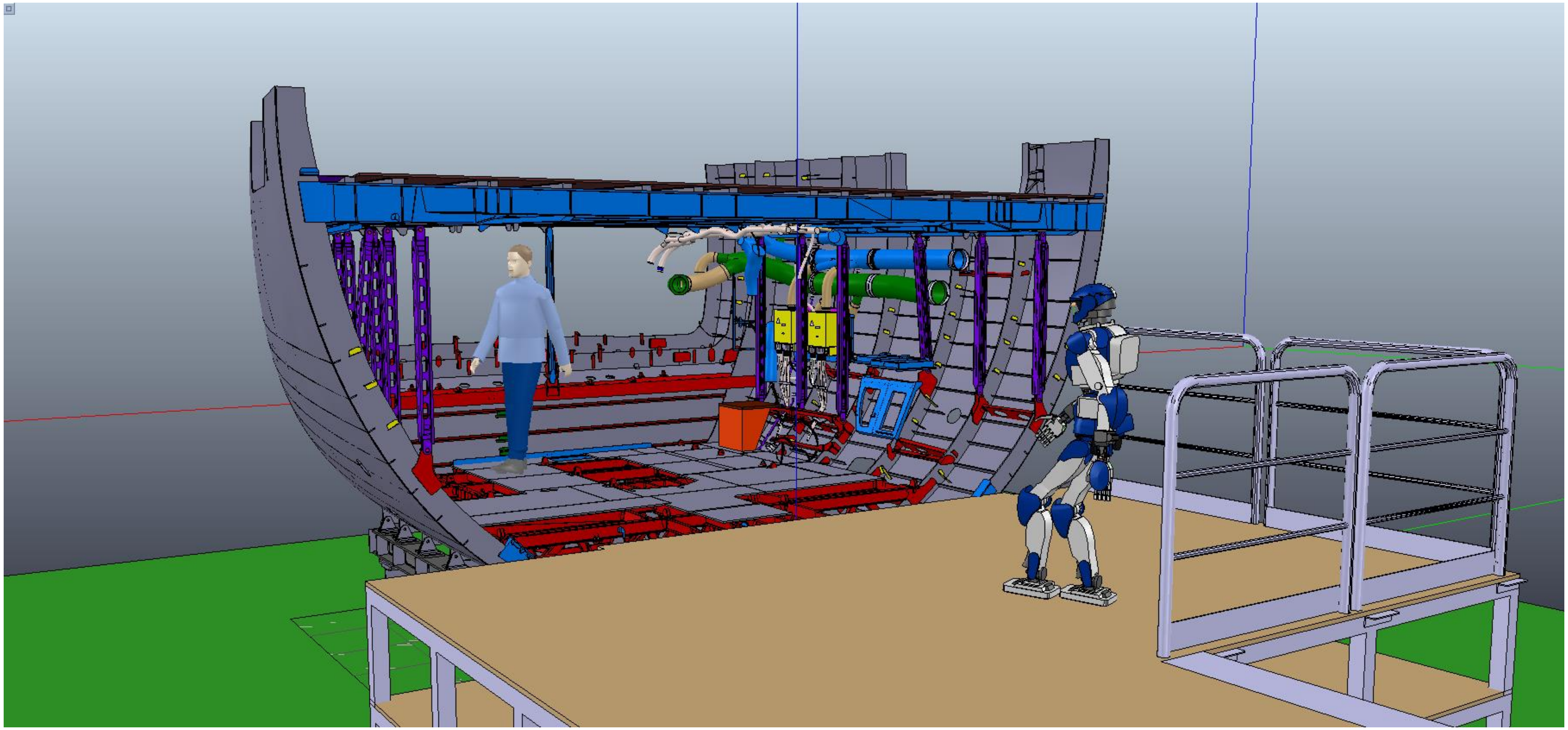


Introduction



Motivations

- Human-robot coexistence
- Robots and human workers share their workspace
- We need to guarantee **safety** for both humans and robots



Topics

- Model Predictive Control for gait generation
  - Linear Inverted Pendulum model
  - MPC scheme
  - Stability constraint
- Real-time pursuit-evasion between humanoid robots
  - Constant velocity moving obstacles
  - Changing velocity obstacles (pursuer)
  - Pursuit-evasion among fixed obstacles

Model Predictive Control for Gait Generation

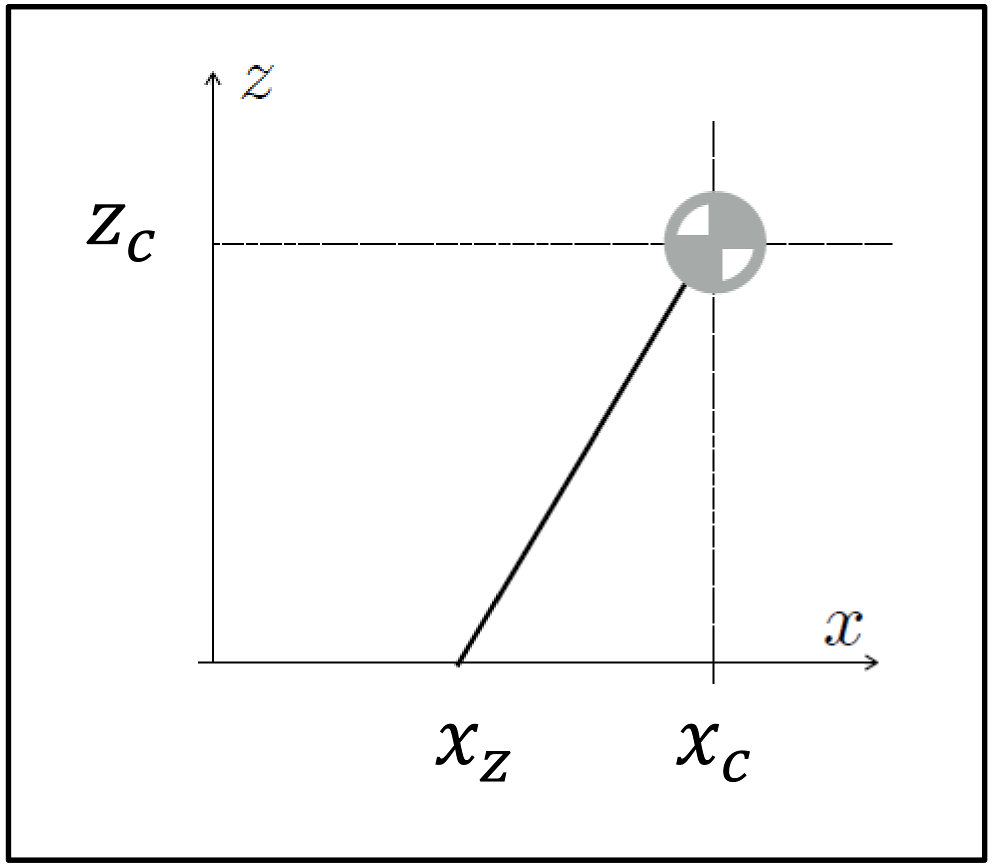
Linear Inverted Pendulum model

The dynamics of the humanoid can be approximated to a Linear Inverted Pendulum (LIP)

$$x_z = x_c - \frac{1}{\omega^2}\ddot{x}_c$$

where $\omega = \sqrt{g/h_{CoM}}$, or as a state-space representation

$$\begin{pmatrix} \dot{x}_c \\ \ddot{x}_c \end{pmatrix} = \begin{pmatrix} 0 & 1 \\ \omega^2 & 0 \end{pmatrix} \begin{pmatrix} x_c \\ \dot{x}_c \end{pmatrix} + \begin{pmatrix} 0 \\ -\omega^2 \end{pmatrix} x_z$$

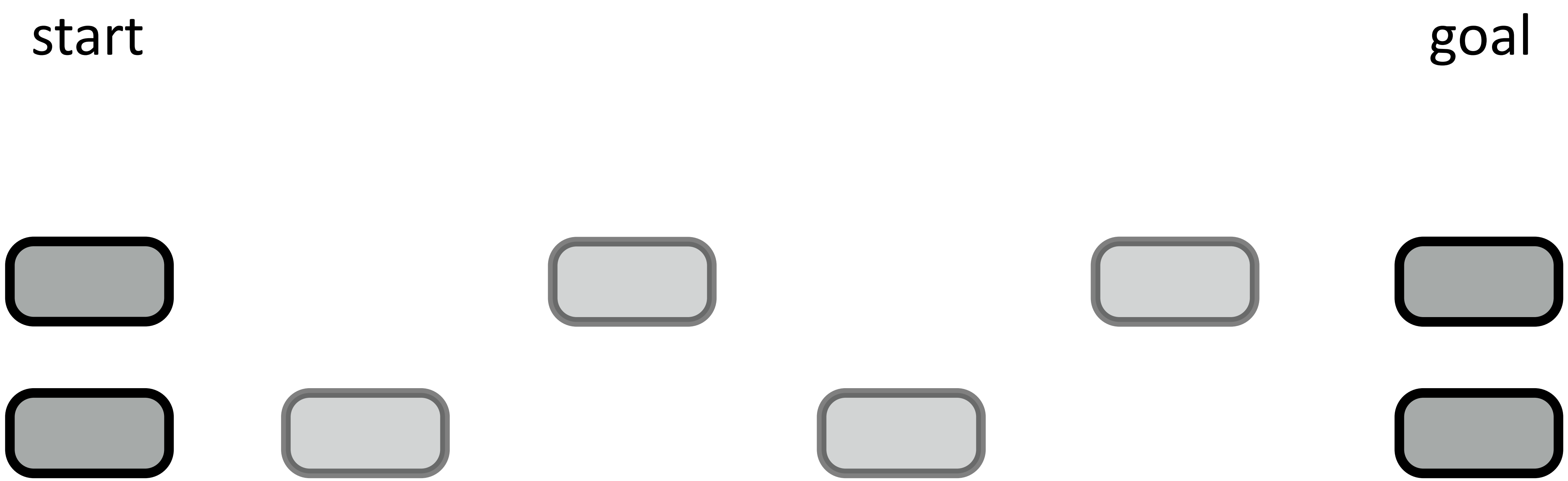


and has two associated modes, of which one is stable and the other is unstable

ZMP-based gait generation

Strategy: keep the ZMP inside the support polygon

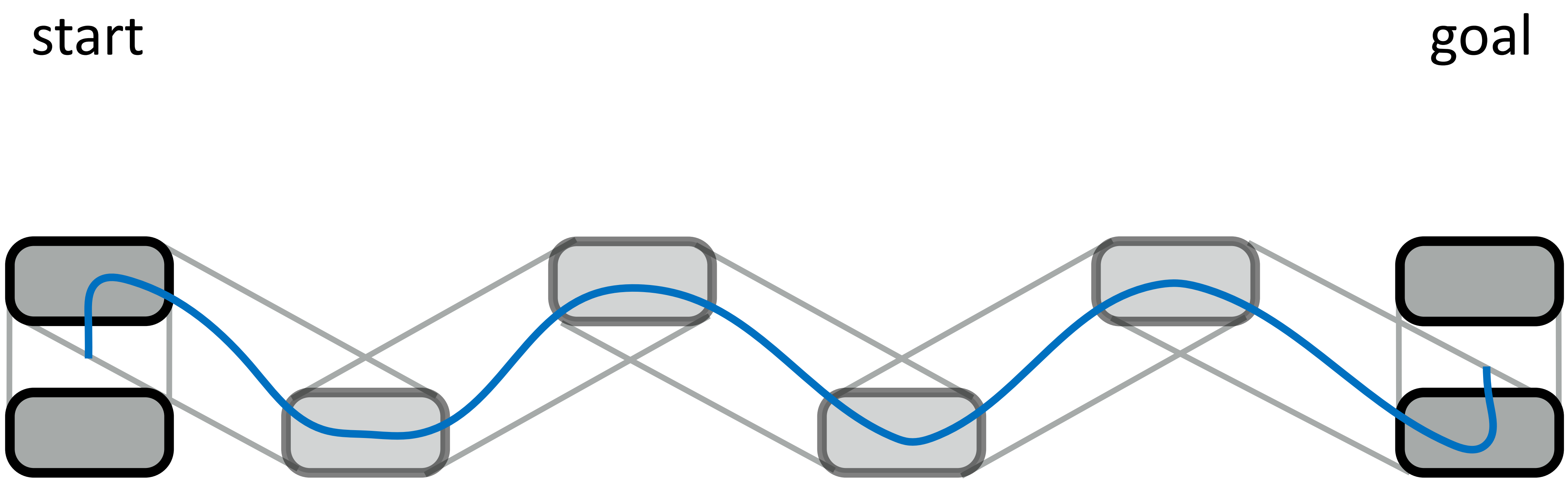
1. Plan the footsteps



ZMP-based gait generation

Strategy: keep the ZMP inside the support polygon

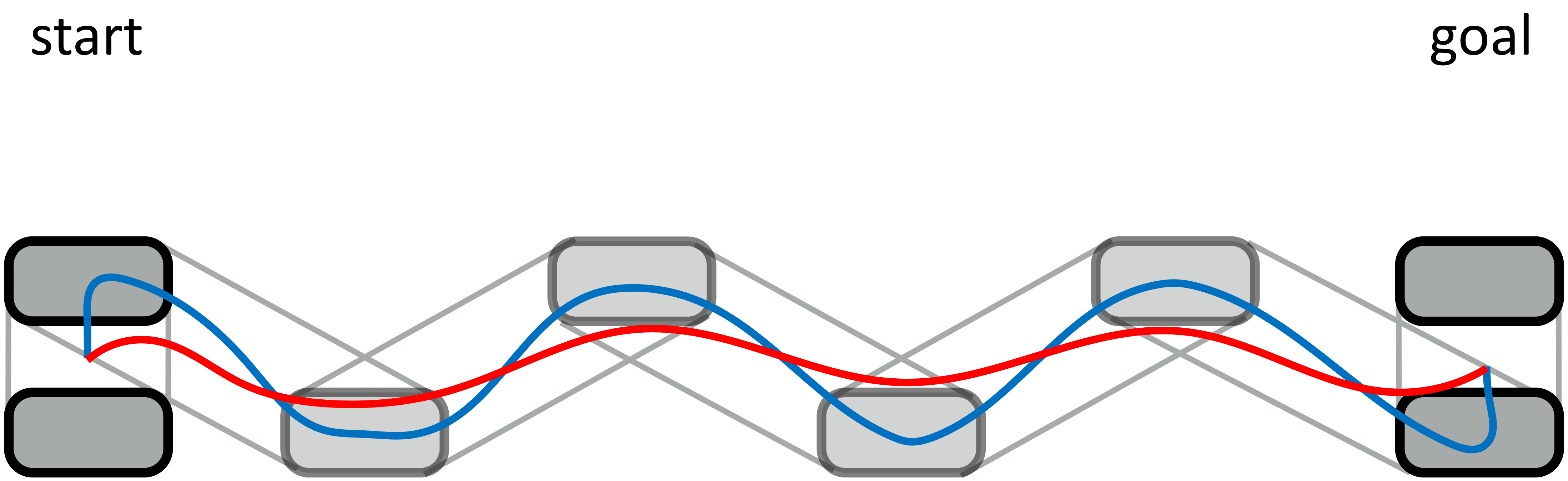
2. Plan a ZMP trajectory that is always inside the support polygon



ZMP-based gait generation

Strategy: keep the ZMP inside the support polygon

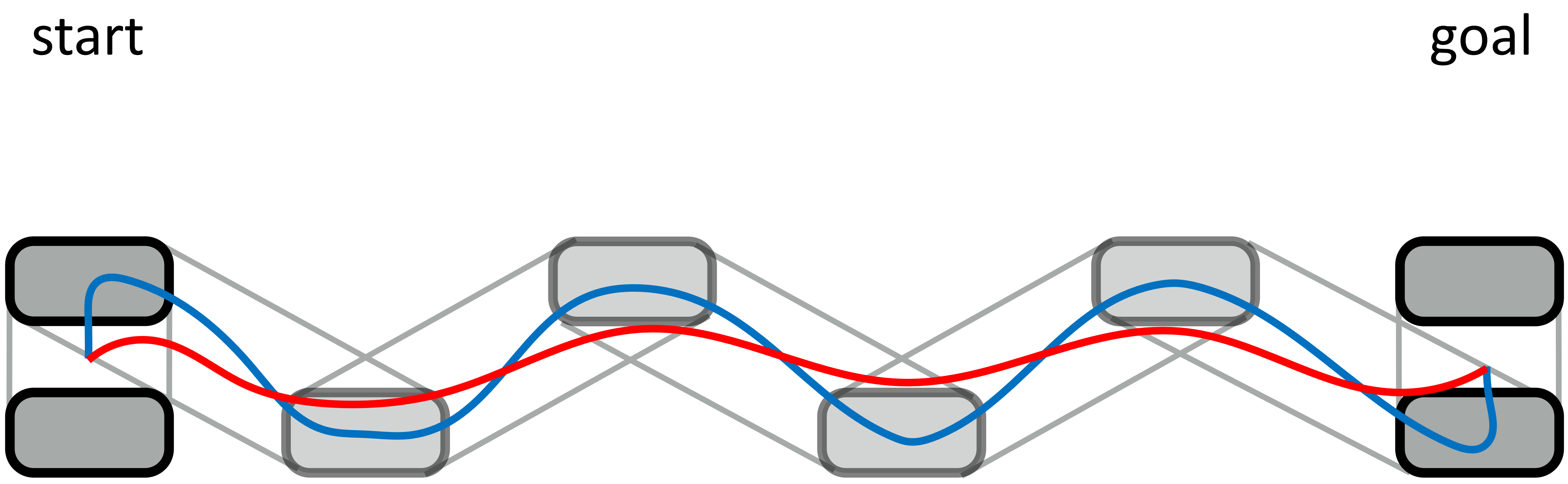
3. Compute a CoM trajectory such that the ZMP moves as planned



ZMP-based gait generation

Strategy: keep the ZMP inside the support polygon

4. Track the CoM trajectory



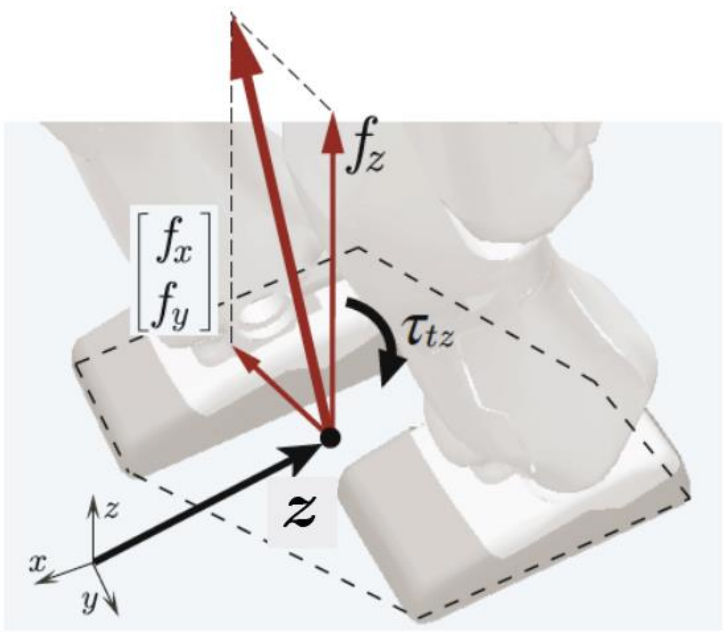
ZMP-based gait generation

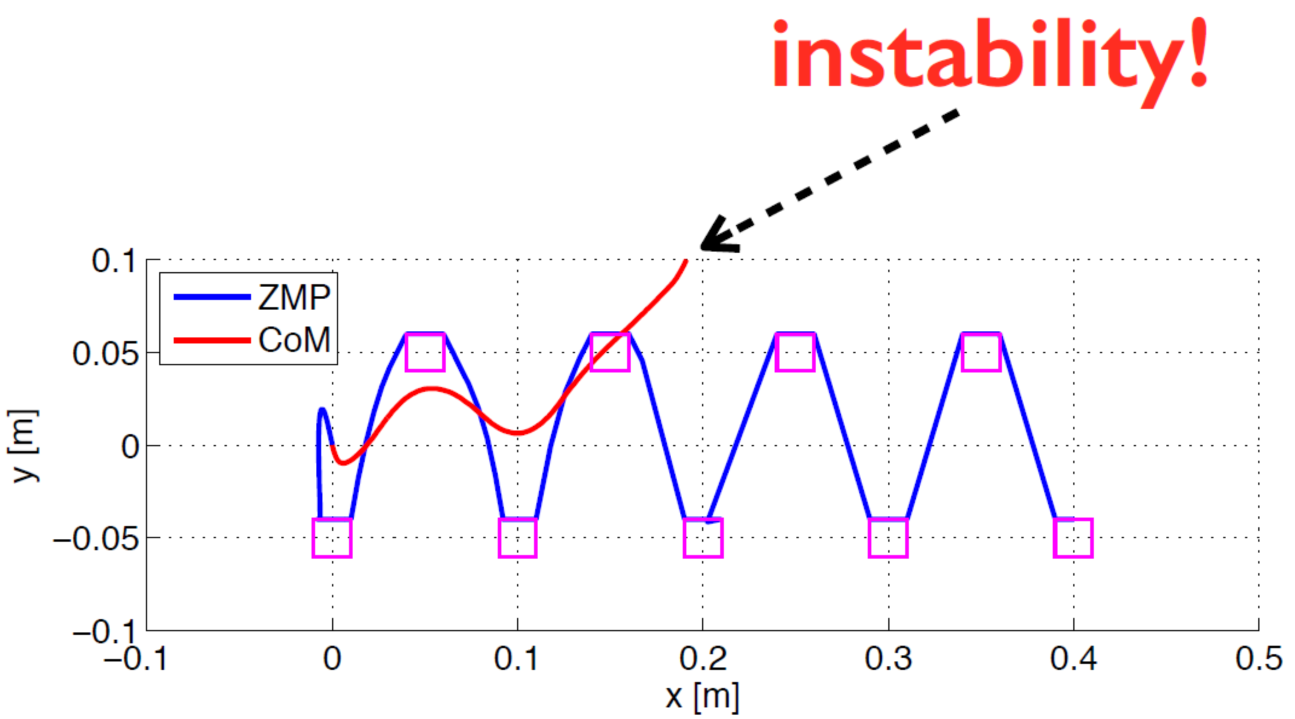
In order to ensure balance we need to keep the ZMP within the support polygon

The LIP has a positive eigenvalue which pertains to an unstable mode

This means that we could have a diverging CoM trajectory even if the ZMP is at all times within the support polygon

balance ≠ stability



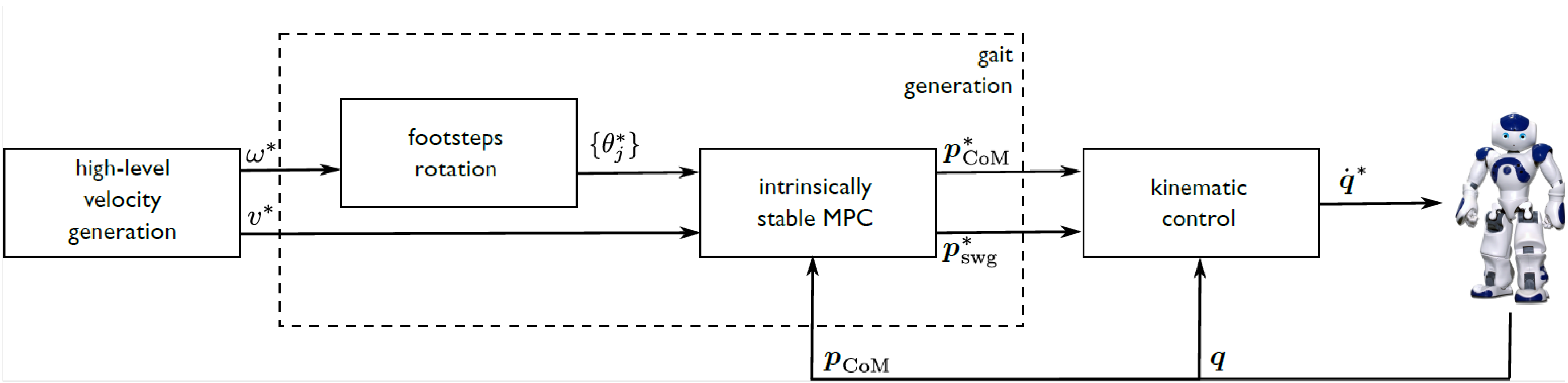


Model Predictive Control

A widely adopted approach to solving the balance problem employs Model Predictive Control (MPC)

MPC can be used to generate a CoM trajectory, which can be tracked with standard Kinematic Control

An important feature of MPC is that it allows to impose constraints



Model Predictive Control

MPC is a form of real-time optimal control

At each iteration we compute the next input by optimizing over a short prediction horizon T_h

- At the k-th iteration look for an optimal control sequence $u_k, u_{k+1}, \dots, u_{k+N}$ over the prediction horizon
- Apply the first control input u_k, then shift the prediction window forward
- Optimize again, this time from $k+1$ to $k+1+N$

Shorter prediction horizons yield less optimality but faster computation

Model Predictive Control

The problem can be formulated as a minimization of a quadratic cost function

$$\min \bar{u}^T H \bar{u} + f^T \bar{u}$$

for a linear system ($\bar{u}$ is a vector containing the next N control inputs), subject to linear constraints

$$A_i u \leq b_i$$
$$A_e u = b_e$$

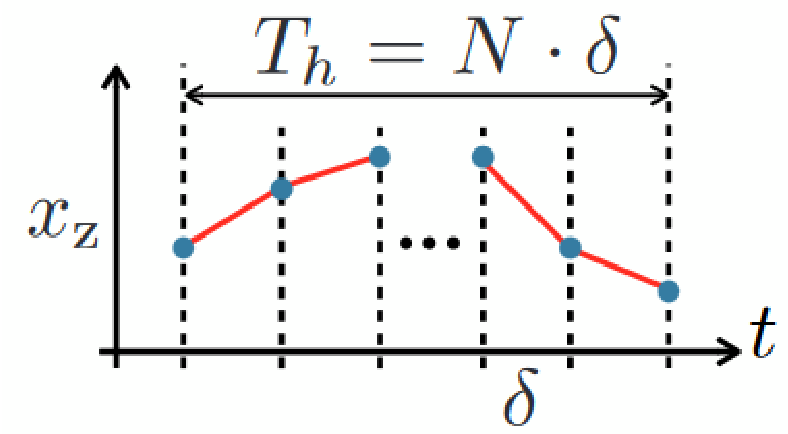
This kind of optimization problem can be solved very efficiently through Quadratic Programming (QP)

In particular, the inequality constraints are very suited for keeping the ZMP within the support polygon

This allows for a robust solution, because we are not tracking any specific trajectory, any solution that satisfies the constraints is fine

Model Predictive Control

Our control variable is the ZMP velocity, assumed piecewise constant with timestep δ



At this point we assume that the footsteps are preassigned

We look for a ZMP trajectory that minimizes the quadratic cost function:

$$\left|\dot{X}_z^k\right|^2 + \left|\dot{Y}_z^k\right|^2$$

under the following constraints:

- ZMP is always inside the support polygon (balance constraint)
- CoM trajectory is bounded w.r.t. the ZMP (stability constraint)

Balance constraints

To maintain balance the ZMP has to be at all times within the support polygon

At the predicted instant $t_k + i\delta$ we apply the constraint

$$x_f^j - \frac{1}{2}s \leq \boxed{x_z^k + \delta \sum_{l=k+1}^{k+i} \dot{x}_z^l} \leq x_f^j + \frac{1}{2}s$$

predicted ZMP

x_f^j is the j-th foot position, and s is the size of the robot foot

Stability constraint

The LIP model has an unstable mode: it is possible for the ZMP to be within the support polygon while the CoM is diverging

Solution: every bounded CoM trajectory $x_c(t)$ is constrained to the generated ZMP $x_z(t)$ by this relation (t_k is the current time)

$$x_c(t_k) + \frac{1}{\omega}\,\dot{x}_c(t_k) = \omega \int_{t_k}^{\infty} e^{-\omega(\tau - t_k)}\, x_z(\tau) d\tau$$

Depends upon all future values of the ZMP

Stability constraint

Since we are making a prediction, we can impose this relation as a constraint on such prediction, at least up to T_h

The walking gait has a tendency to periodicity, we compute the integral after T_h by infinitely replicating the control inputs (although there are other possible choices)

Final stability constraint

$$\frac{1}{\omega}\frac{1-e^{-\omega\delta}}{1-e^{-N\omega\delta}}\sum_{i=0}^{N-1} e^{-i\omega\delta}\dot{x}_z(t_{k+i}) = x_c(t_k) + \frac{\dot{x}_c(t_k)}{\omega} - x_z(t_k)$$

Automatic footstep placement

No prior knowledge of the footstep positions. The predicted footsteps are included as additional control variables

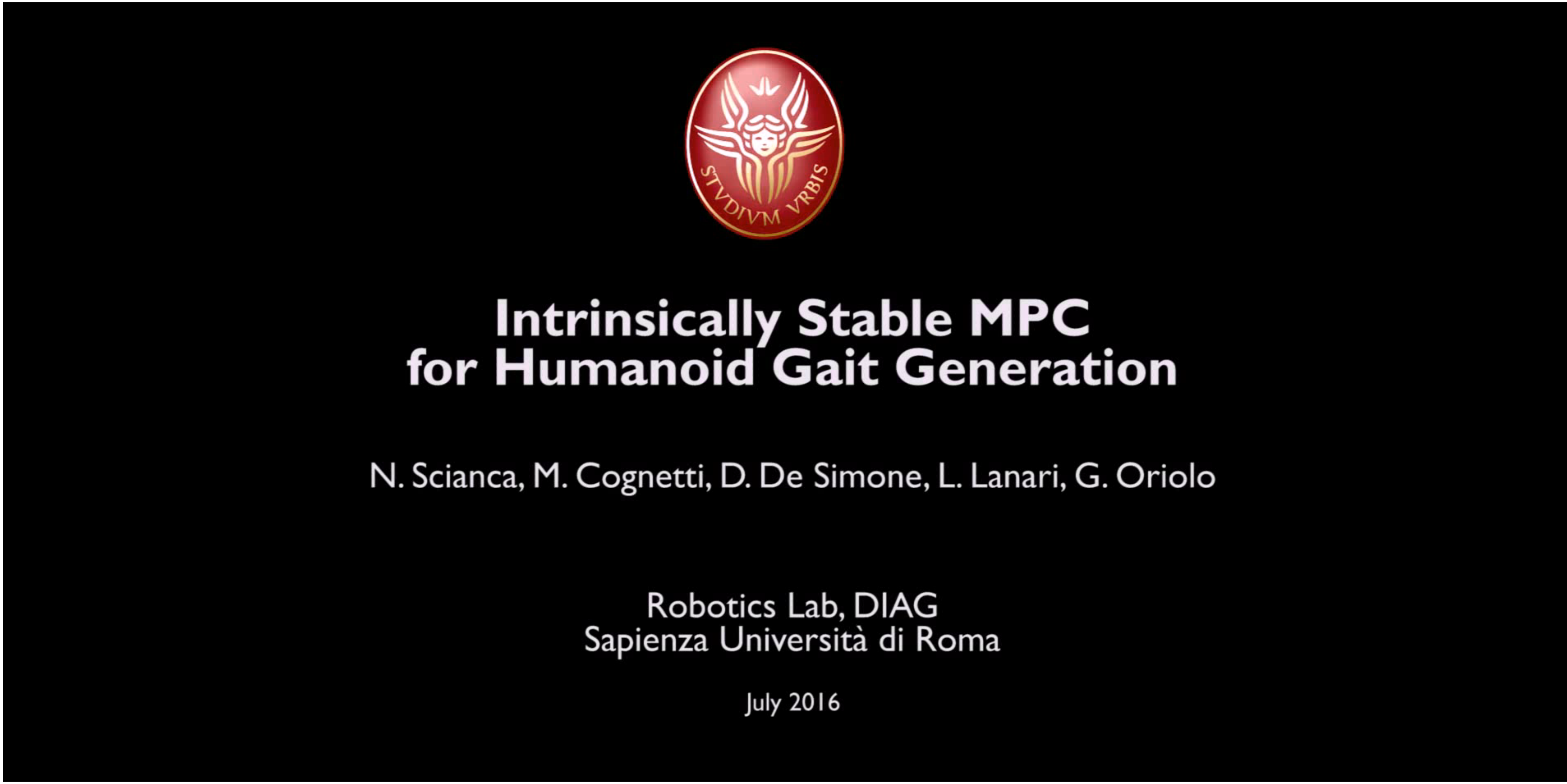
$$\left(\dot{X}_z^k \quad X_f^k \quad \dot{Y}_z^k \quad Y_f^k\right)$$

The new goal is to track a reference velocity, which is done by adding a term to the cost function

$$\left|\dot{X}_z^k\right|^2 + \left|\dot{Y}_z^k\right|^2 + \left|\dot{X}_c^k - v_{ref}\right|^2$$

- In order to maintain linearity the balance constraints are enforced only in single support phases
- The stability constraint is unchanged (depends only on the predicted ZMP trajectory)
- Additional constraints on the predicted footsteps to ensure feasibility

Simulations



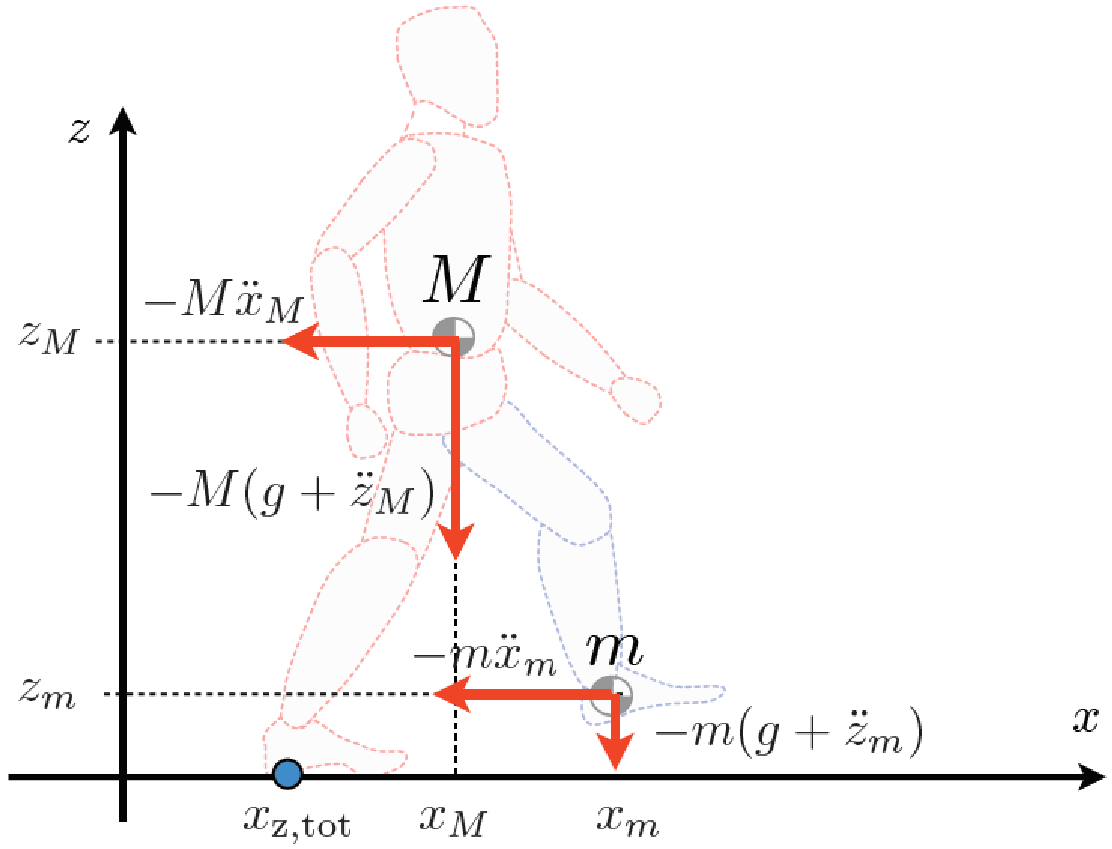
https://youtu.be/hYegqFoeCJc

MPC on a Multi-Mass Model

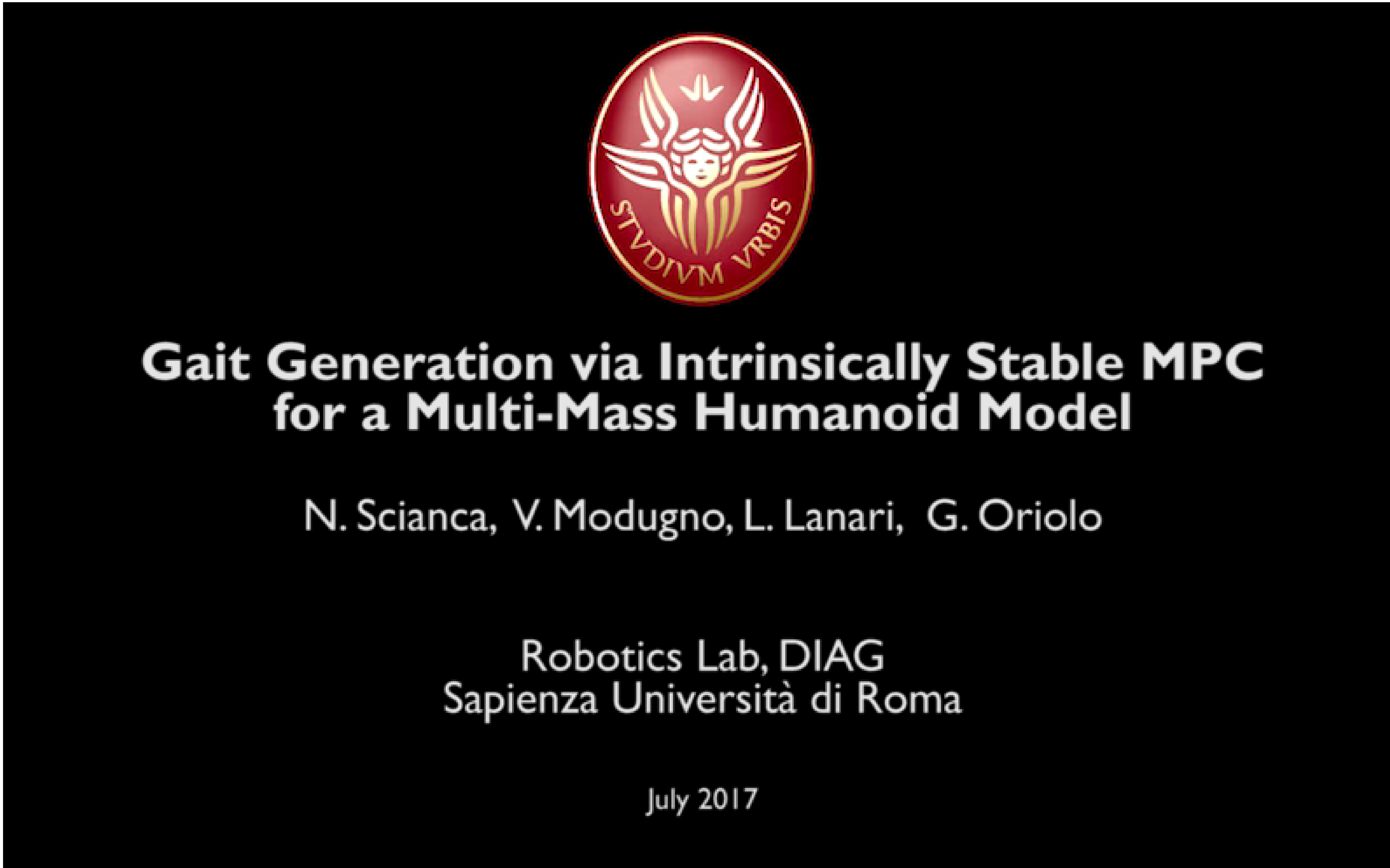
A second mass accounts for the contribution to the ZMP given by the swinging leg

$$x_{z,tot} = \left(1 + \frac{m}{M}\frac{\ddot{z}_m + g}{\ddot{z}_M + g}\right)^{-1}\left(x_{z,M} + \frac{m}{M}\frac{\ddot{z}_m + g}{\ddot{z}_M + g}x_{z,m}\right)$$

- Partially compensates for neglected angular momentum
- Measured ZMP is closer to the nominal prediction
- More robust to uncertainties



Simulations



https://youtu.be/CwpWX2isypk

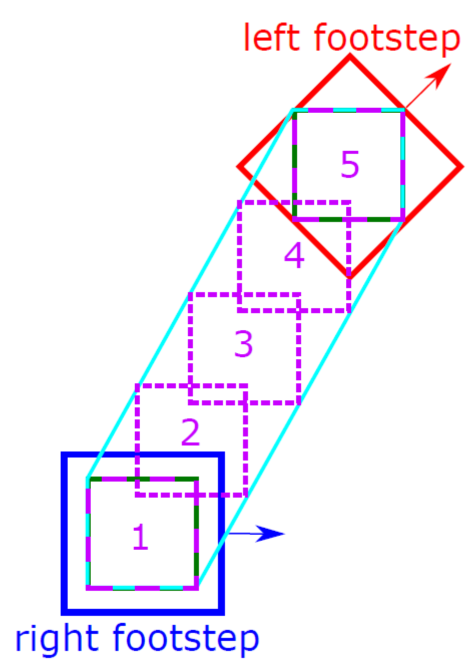
MPC for Walk-To Locomotion

Instead of tracking a reference velocity, we want the cost function to penalize the distance from the goal

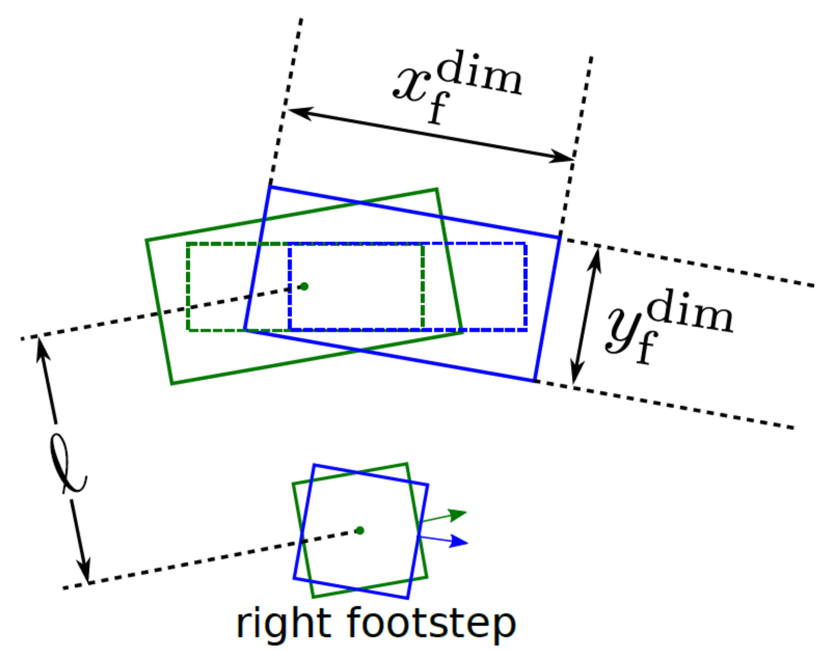
$$|\dot{X}_z^k|^2 + |\dot{Y}_z^k|^2 + \boxed{|X_c^k - x_{goal}|^2 + |Y_c^k - y_{goal}|^2}$$

- Optimization needs to generate also footstep orientations
- Constraints become nonlinear, need to be approximated

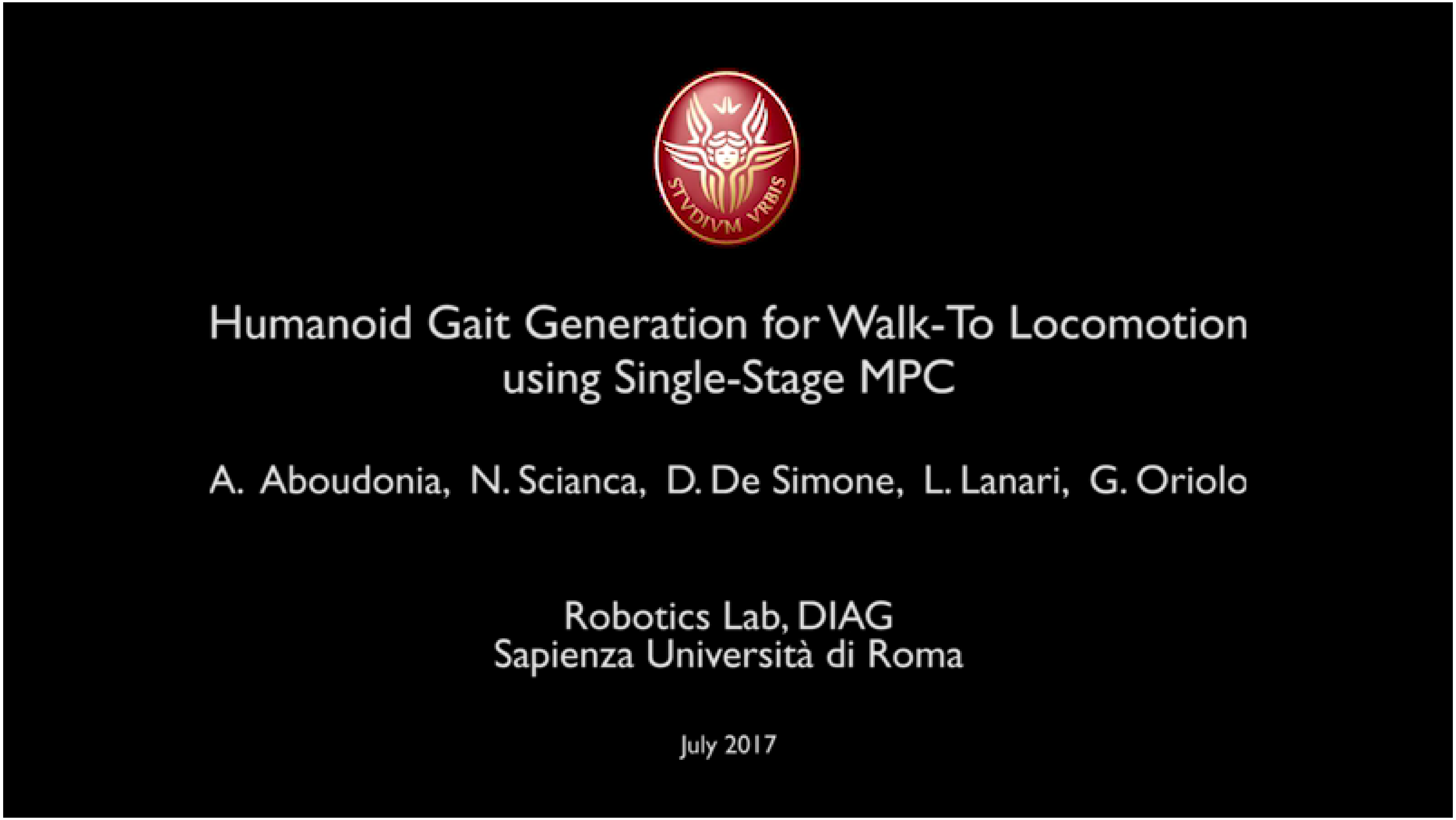
ZMP constraint



footstep placement constraint



Simulations



https://youtu.be/fmmehMItOGw

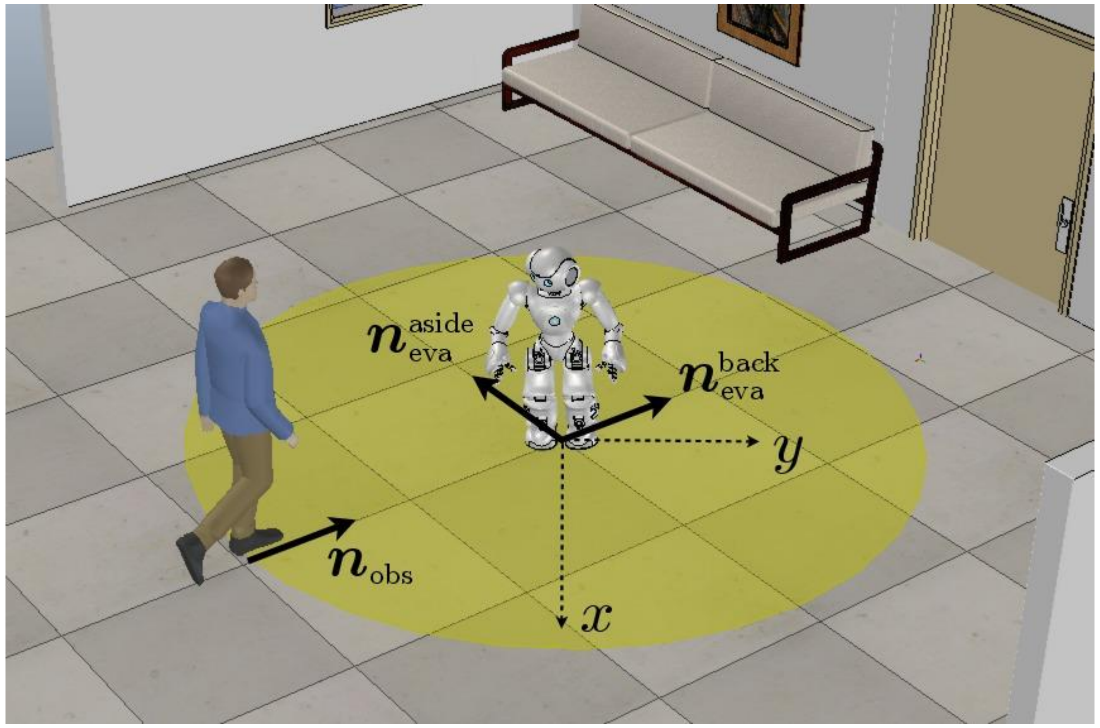
Real-time Pursuit Evasion for Humanoid Robots

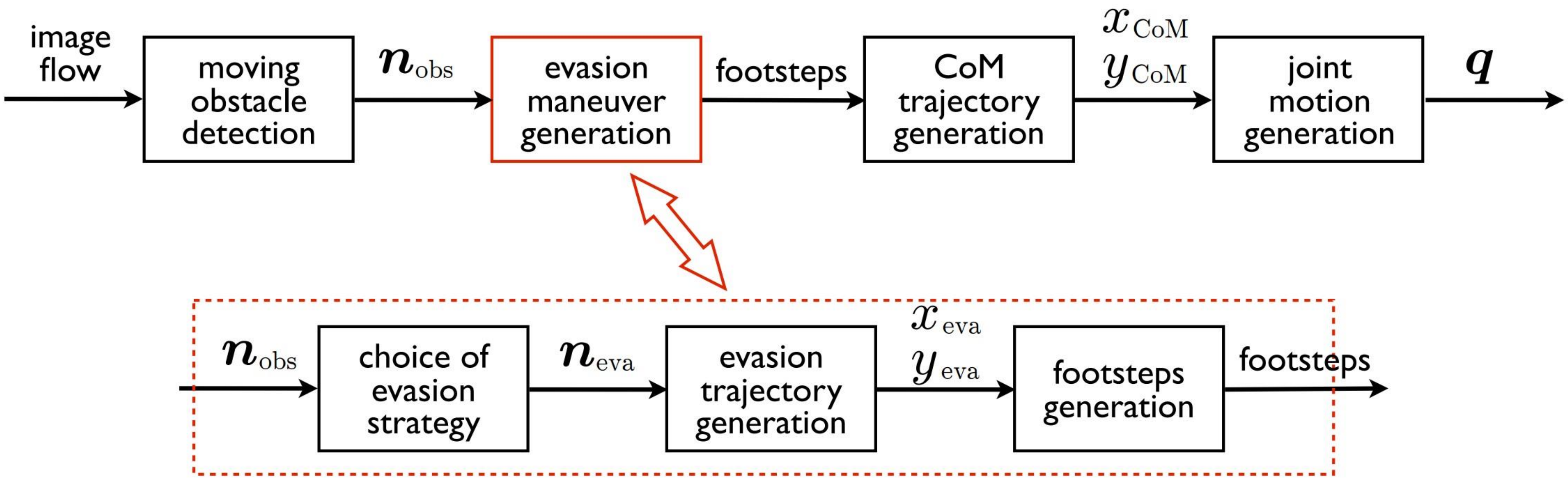
Problem Formulation 1

A humanoid robot is standing in its workspace when a moving obstacle enters its **safety area** moving with constant velocity

An evasive motion must be generated to avoid the collision

Commands should be made available to the robot controller in **real time**

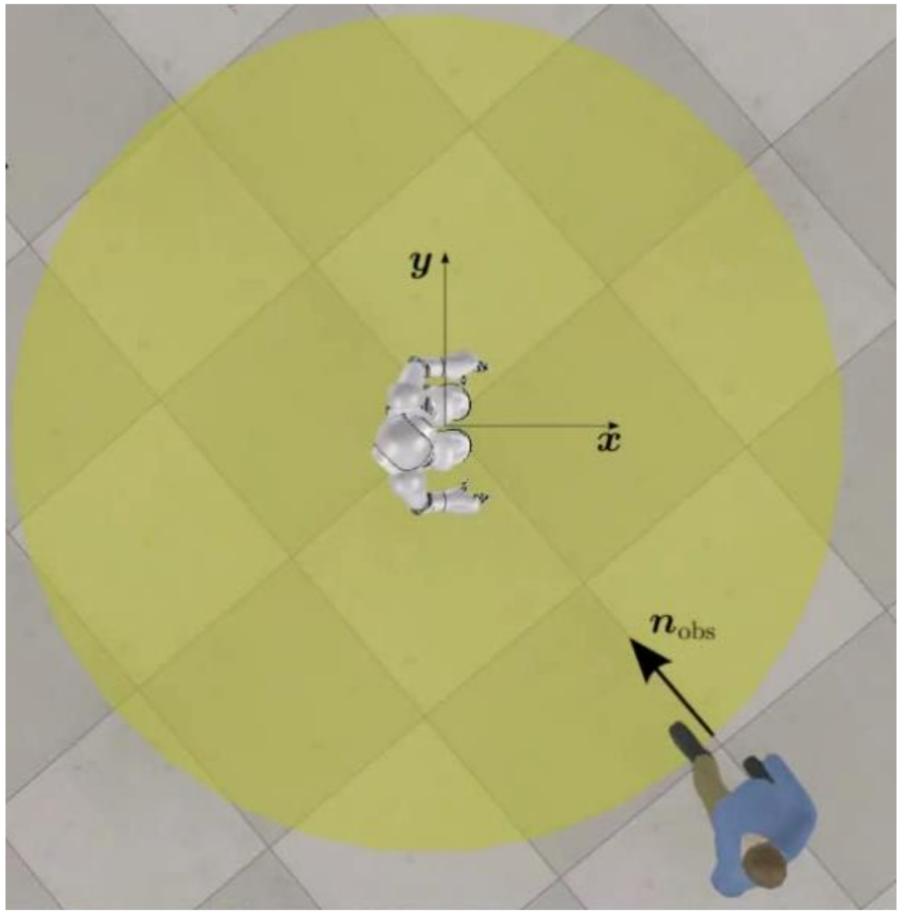




Moving Obstacle Detection

Using a depth camera the obstacle is detected while entering the robot safety region

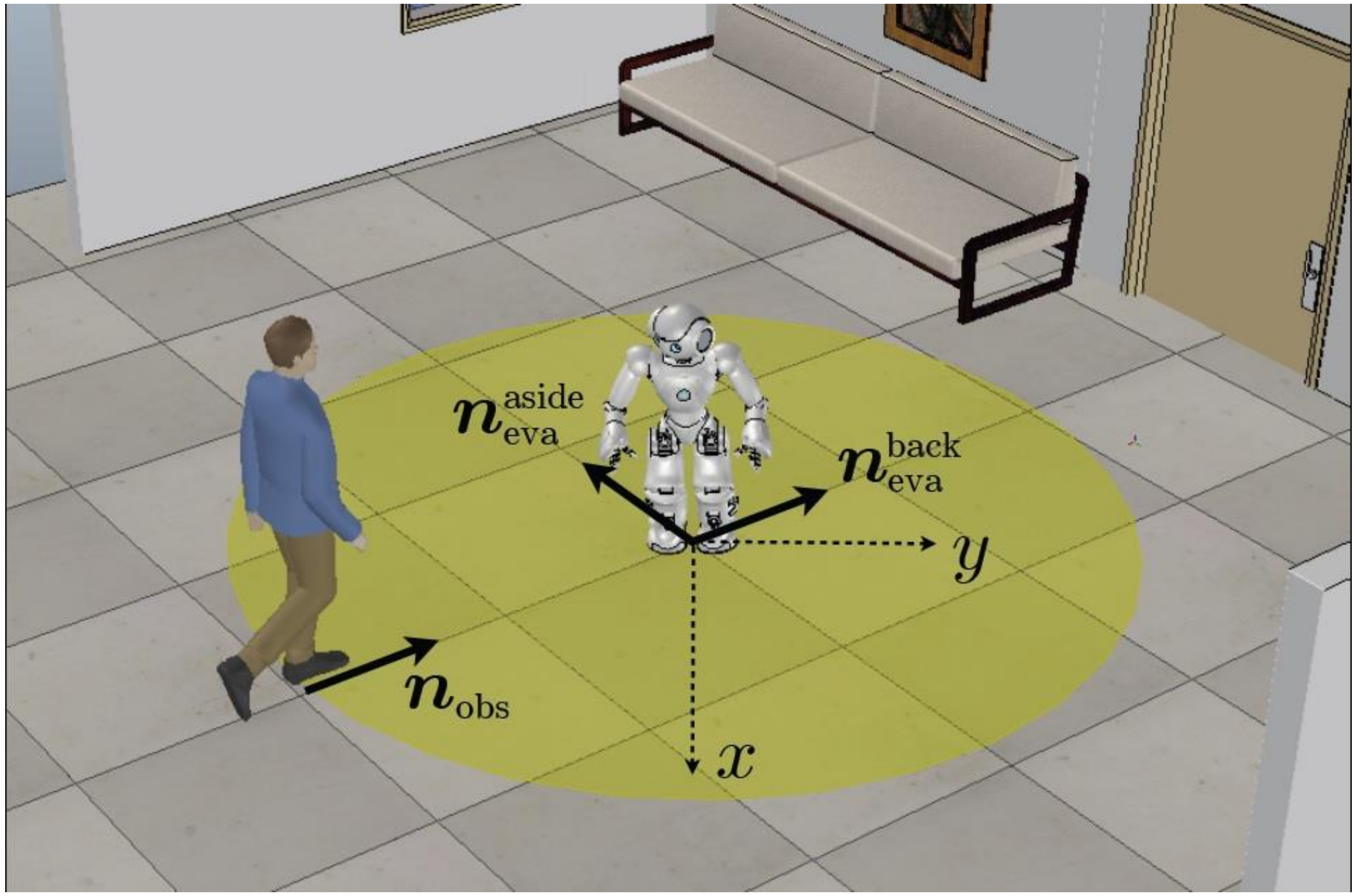
Through the sensor, the approach direction of the obstacle relative to the humanoid is computed



Evasion Maneuver Generation

Two possible evasion strategies:

- **Move Back:** the humanoid aligns with the direction of the incoming obstacle and moves backwards
- **Move Aside:** the humanoid aligns with the direction orthogonal to that of the moving obstacle and moves backwards

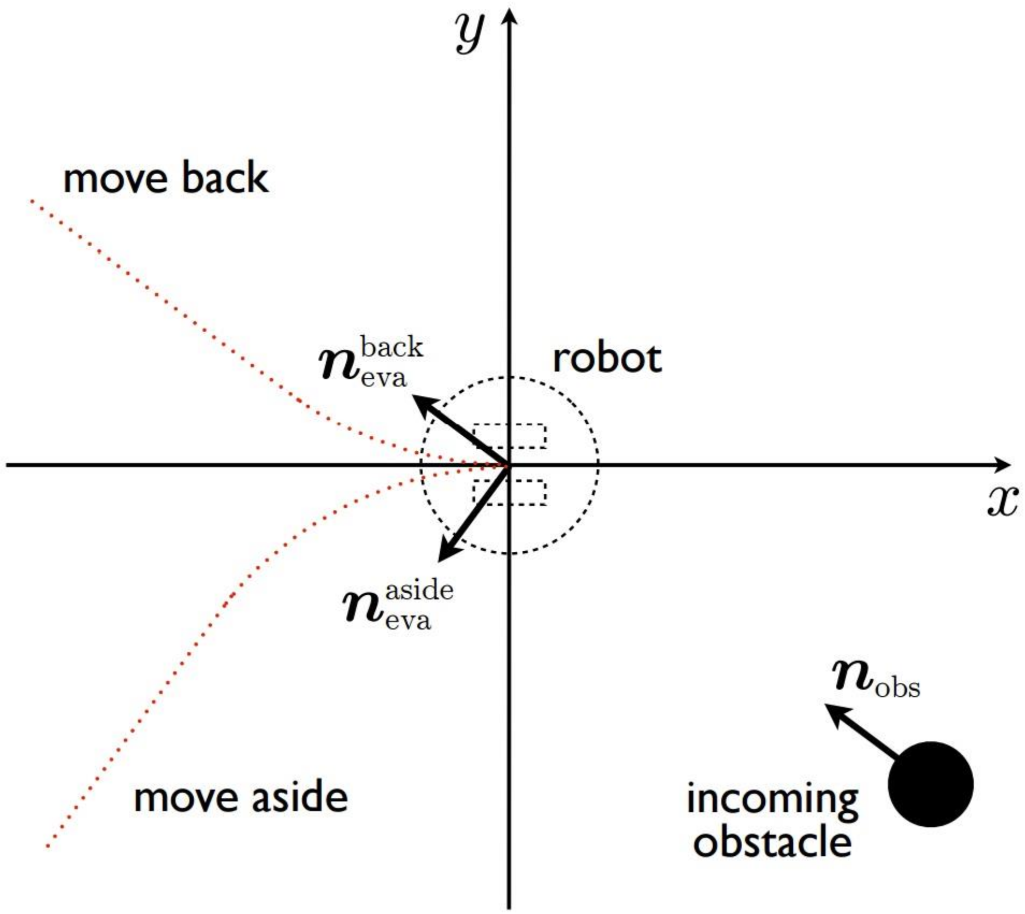


Evasion Maneuver Generation

A controlled unicycle model is used to generate the evasion trajectory

$$\dot{x} = v\cos\theta$$
$$\dot{y} = v\sin\theta$$
$$\dot{\theta} = \omega$$

$$v = \bar{v}$$
$$\omega = k\ sign\ (\theta_{eva} - \theta)$$



Evasion Maneuver Generation

Integration of model equation under the proposed control law provides a closed form for the evasion trajectory

$$
\begin{aligned}
x(t) &= \bar{v}\,\frac{\sin kt}{k} \\
y(t) &= \operatorname{sign}(\theta_{\text{eva}})\,\bar{v}\,\frac{1-\cos kt}{k} \\
\theta(t) &= \operatorname{sign}(\theta_{\text{eva}})\,kt
\end{aligned}
$$

for $t \leq t_s$ and

$$
\begin{aligned}
x(t) &= x(t_s) + \bar{v}(t-t_s)\cos\theta_{\text{eva}} \\
y(t) &= y(t_s) + \bar{v}(t-t_s)\sin\theta_{\text{eva}} \\
\theta(t) &= \theta_{\text{eva}}
\end{aligned}
$$

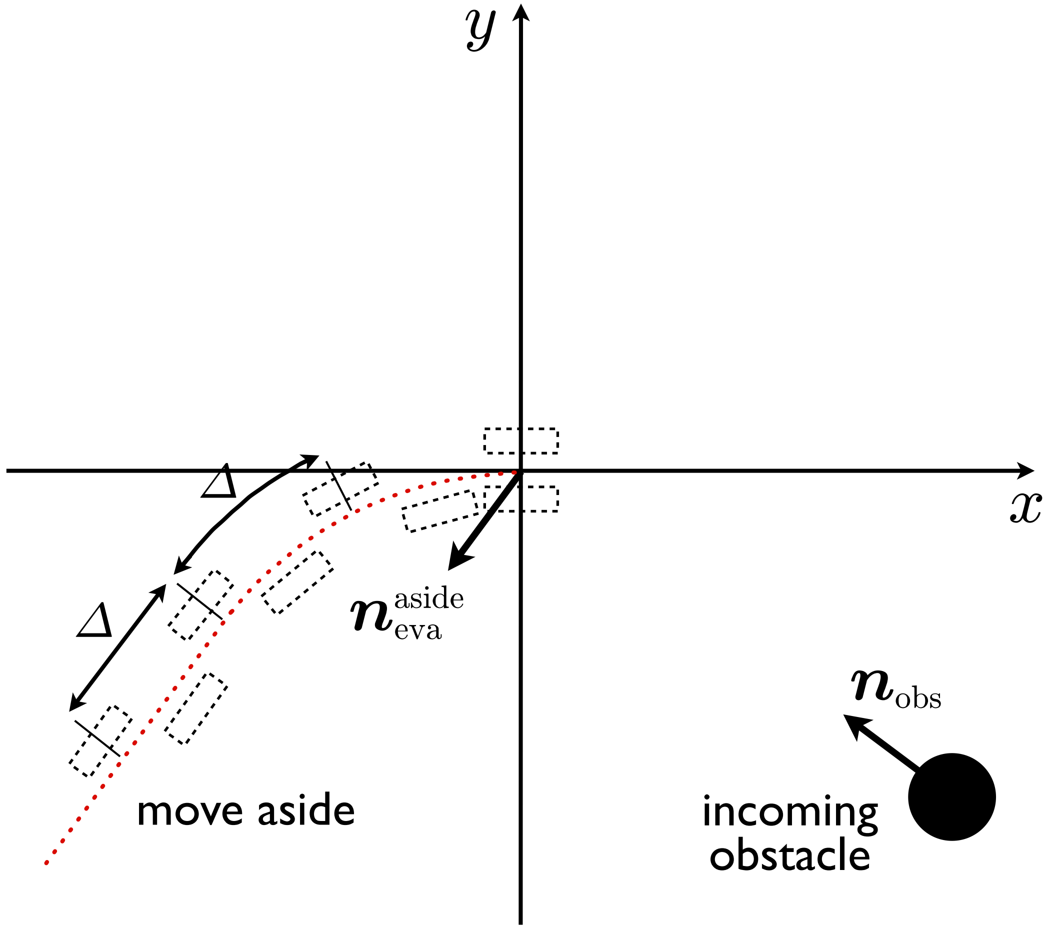
for $t > t_s$

Evasion Maneuver Generation

Once the trajectory is computed, footsteps for the robot are planned to follow the trajectory

The trajectory is sampled using a constant time interval and the coordinates of the footsteps are computed by displacing samples alternatively to the left and right of the trajectory

$$x_{r,k} = x_k + d \sin\theta_k$$
$$y_{r,k} = y_k - d \cos\theta_k$$

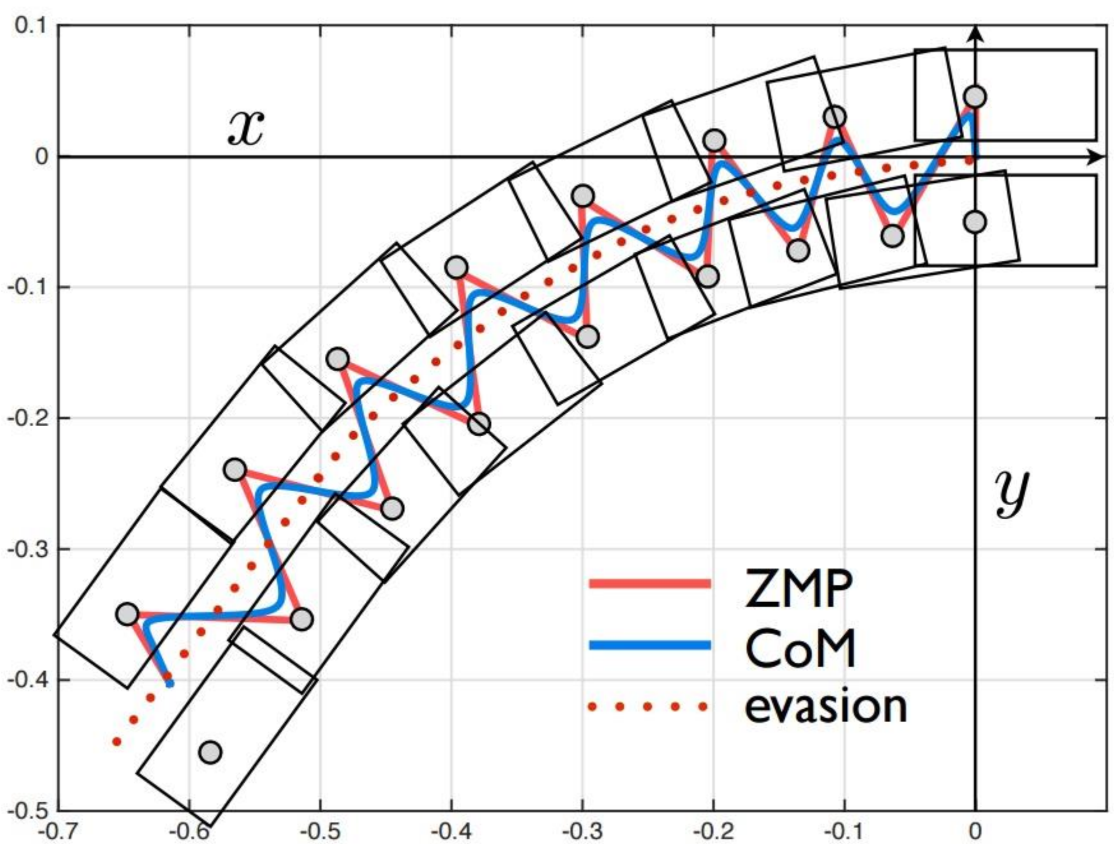


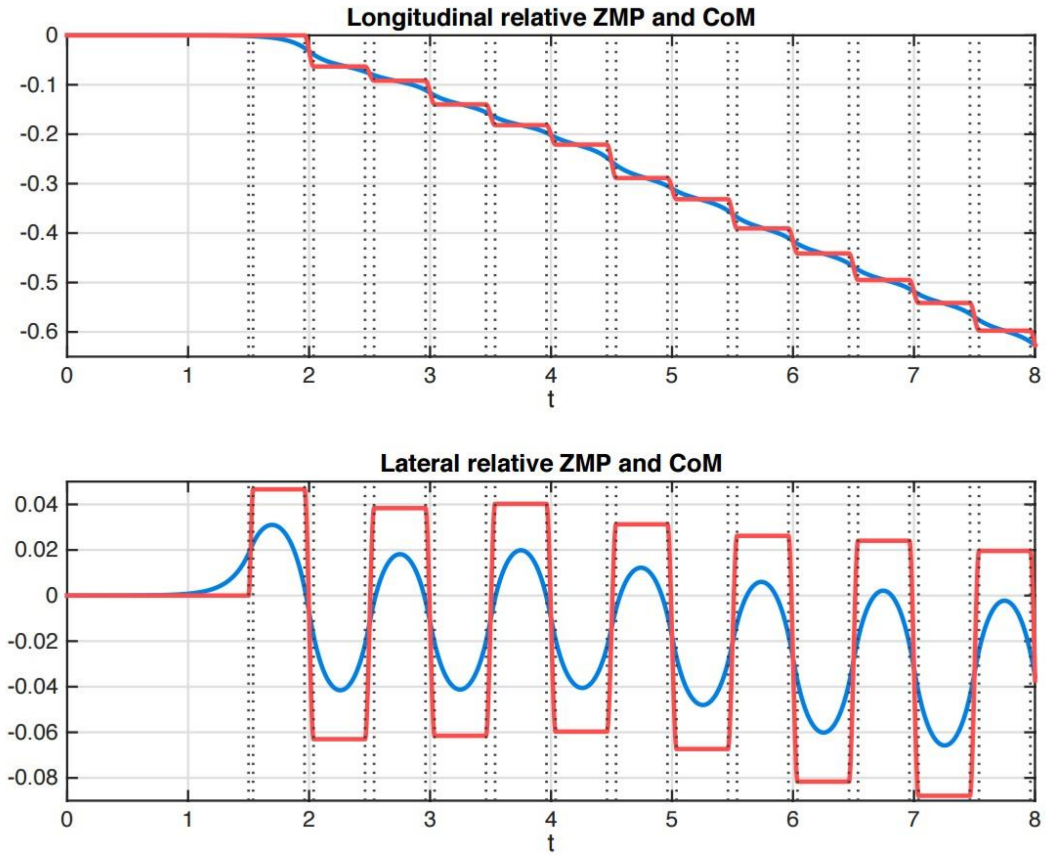
Center of Mass Trajectory Generation

ZMP reference is generated starting from the footsteps

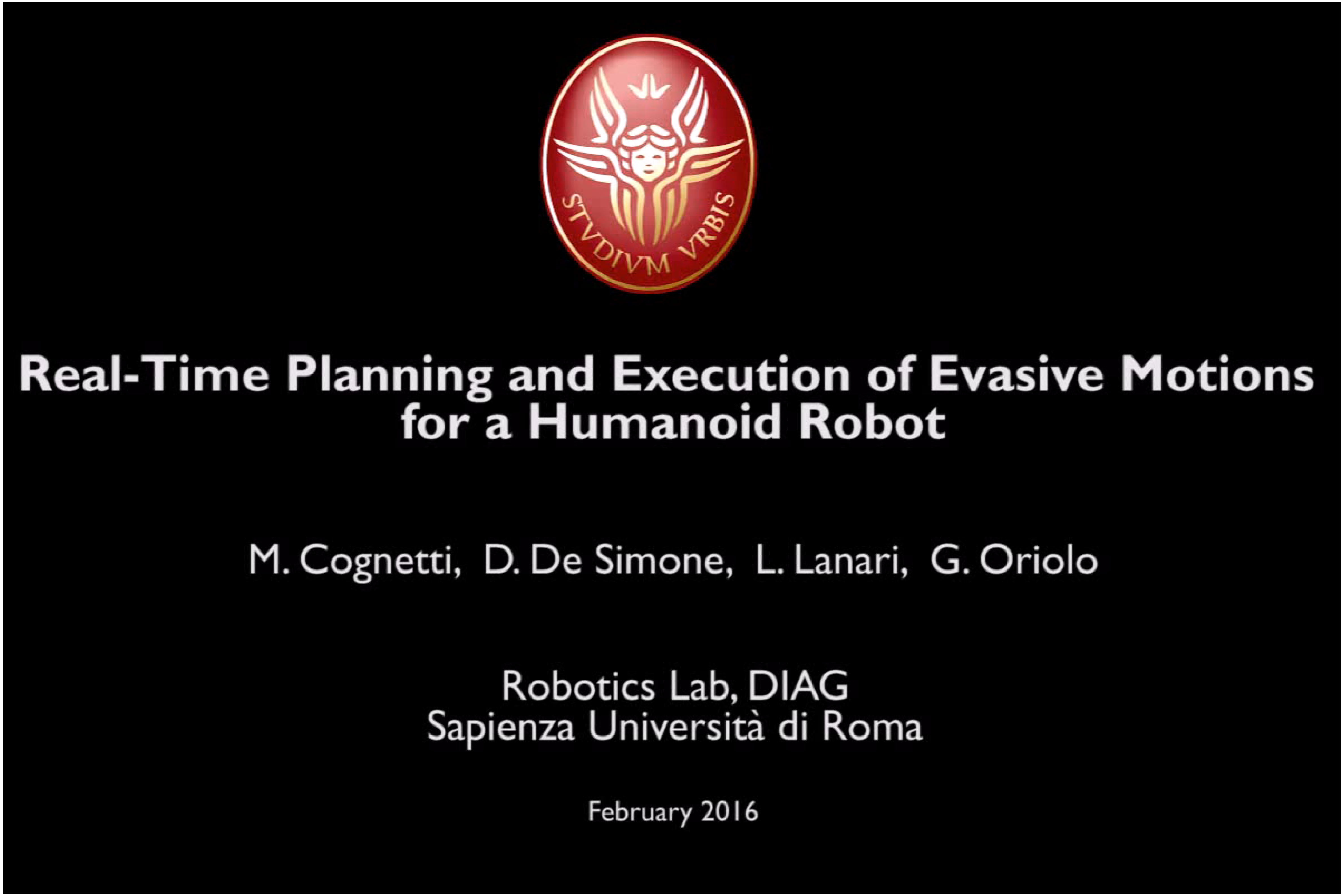
The Center of Mass trajectory is computed with the ZMP as an input

$$x^*_{\mathrm{CoM}}(t) = e^{-\eta t} x_{\mathrm{CoM}}(0) + \frac{x_{\mathrm{s}}(t) - e^{-\eta t} x_{\mathrm{u}}(0) + x_{\mathrm{u}}(t)}{2}$$





Wrapping Up



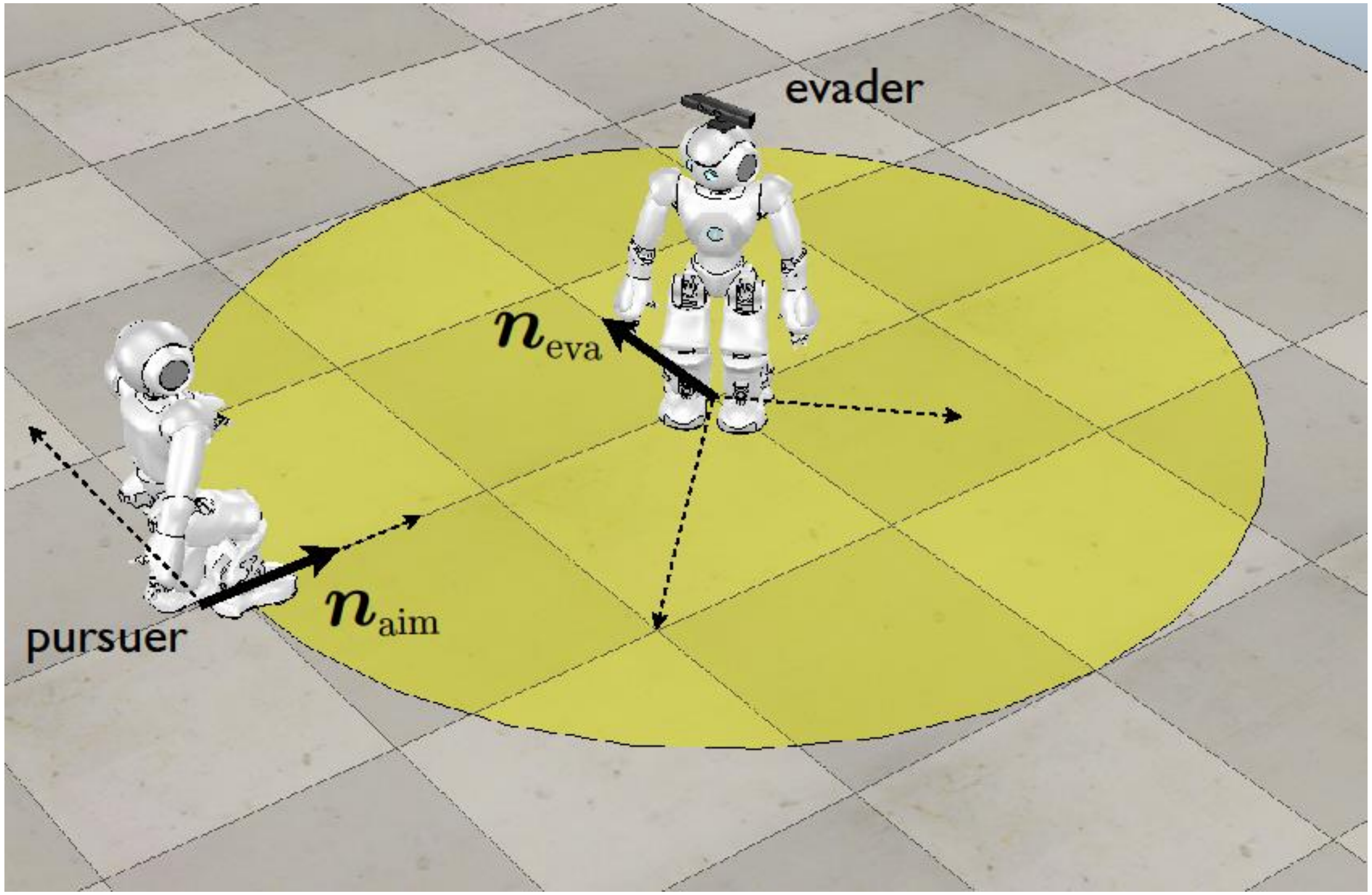
https://youtu.be/mJgJXCTjiYc

Problem Formulation 2

Removed the assumption of constant moving obstacle velocity

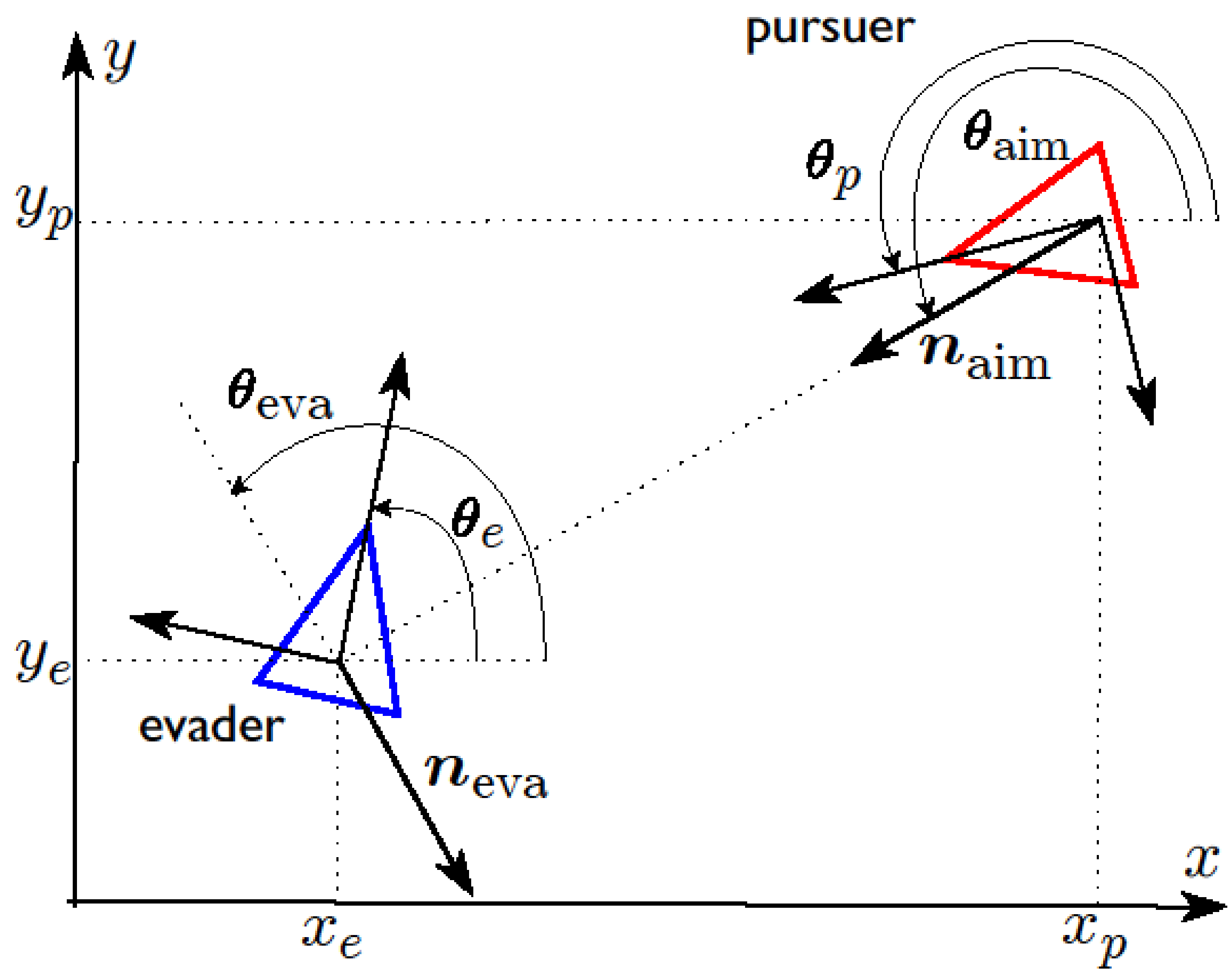
The moving obstacle (pursuer) actively tries to intercept the other robot (evader)

Commands should be replanned in **real time**



Pursuit-Evasion Between Unicycles

Unicycles used as template model for real-time trajectory generation



Pursuit-Evasion Between Unicycles

The two unicycles are controlled with two different control laws

$$\dot{x}_p = v_p \cos\theta_p$$
$$\dot{y}_p = v_p \sin\theta_p$$
$$\dot{\theta}_p = \omega_p$$

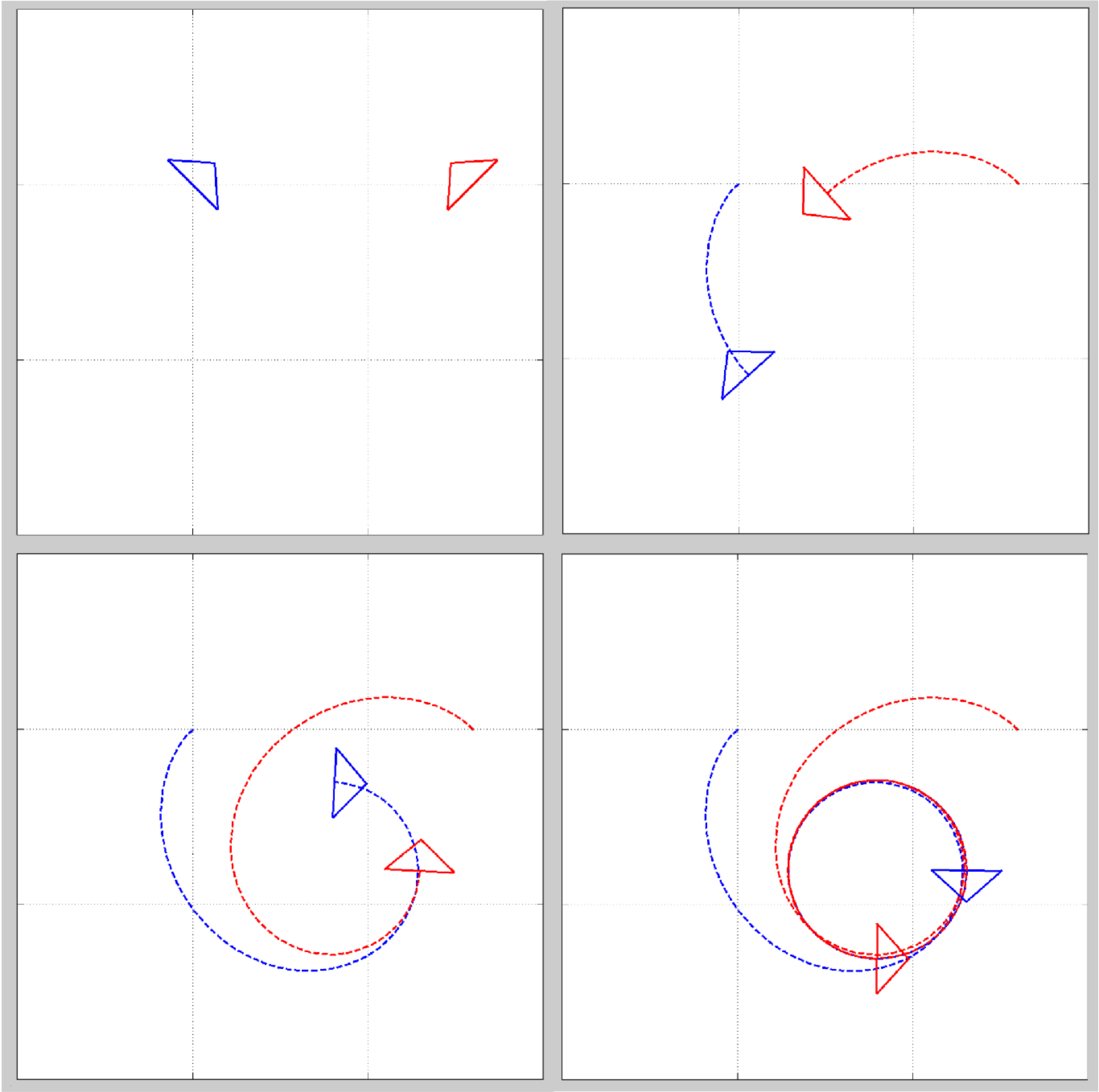
$$\dot{x}_e = v_e \cos\theta_e$$
$$\dot{y}_e = v_e \sin\theta_e$$
$$\dot{\theta}_e = \omega_e$$

$$v_p = \bar{v}$$
$$\omega_p = k\left(\theta_{aim} - \theta_p\right)$$

$$v_e = -\bar{v}$$
$$\omega_e = k(\theta_{eva} - \theta_e)$$

Pursuit-Evasion Between Unicycles

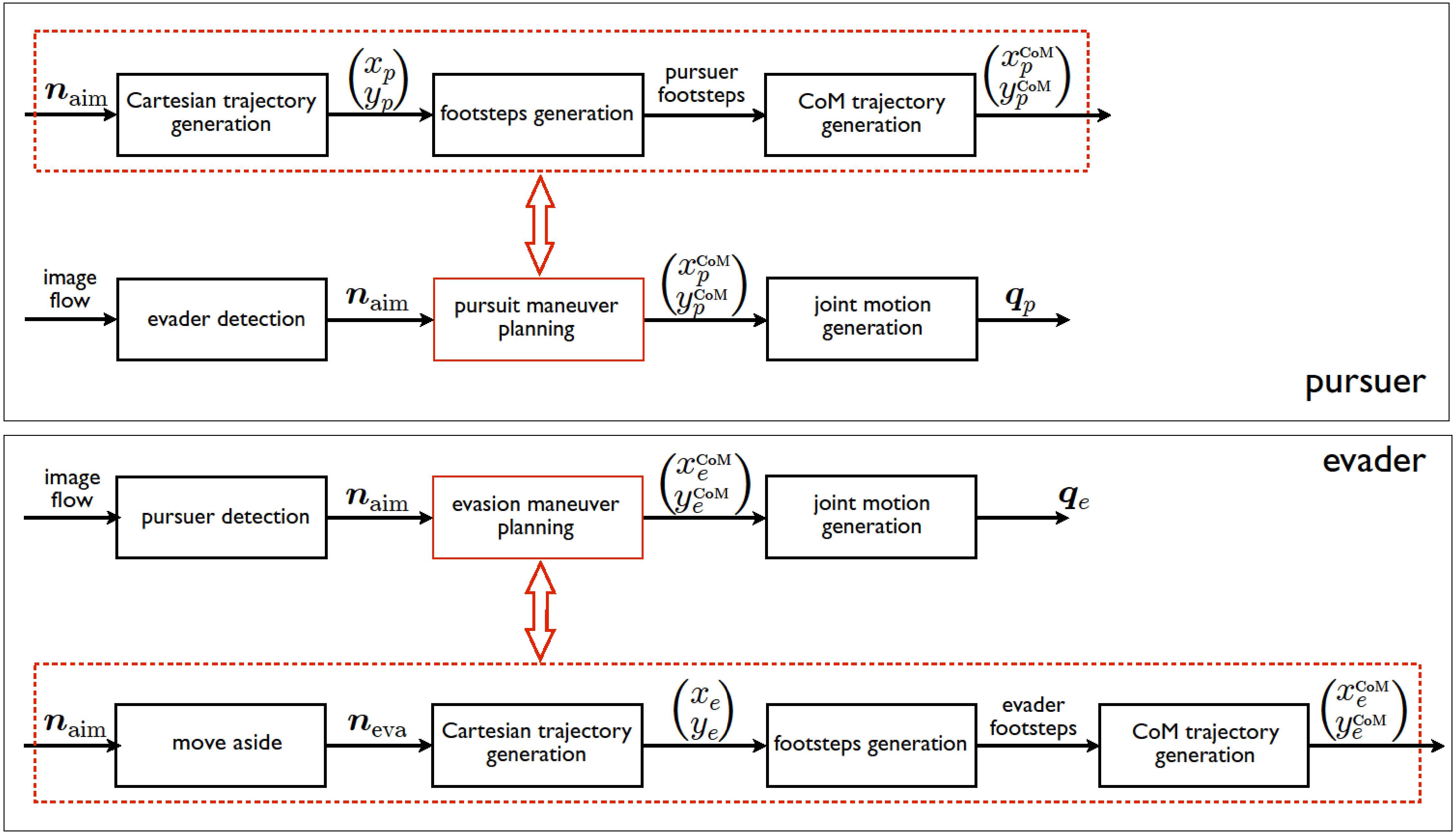
Simulations showed interesting asymptotic behaviors of the Pursuer-Evader system



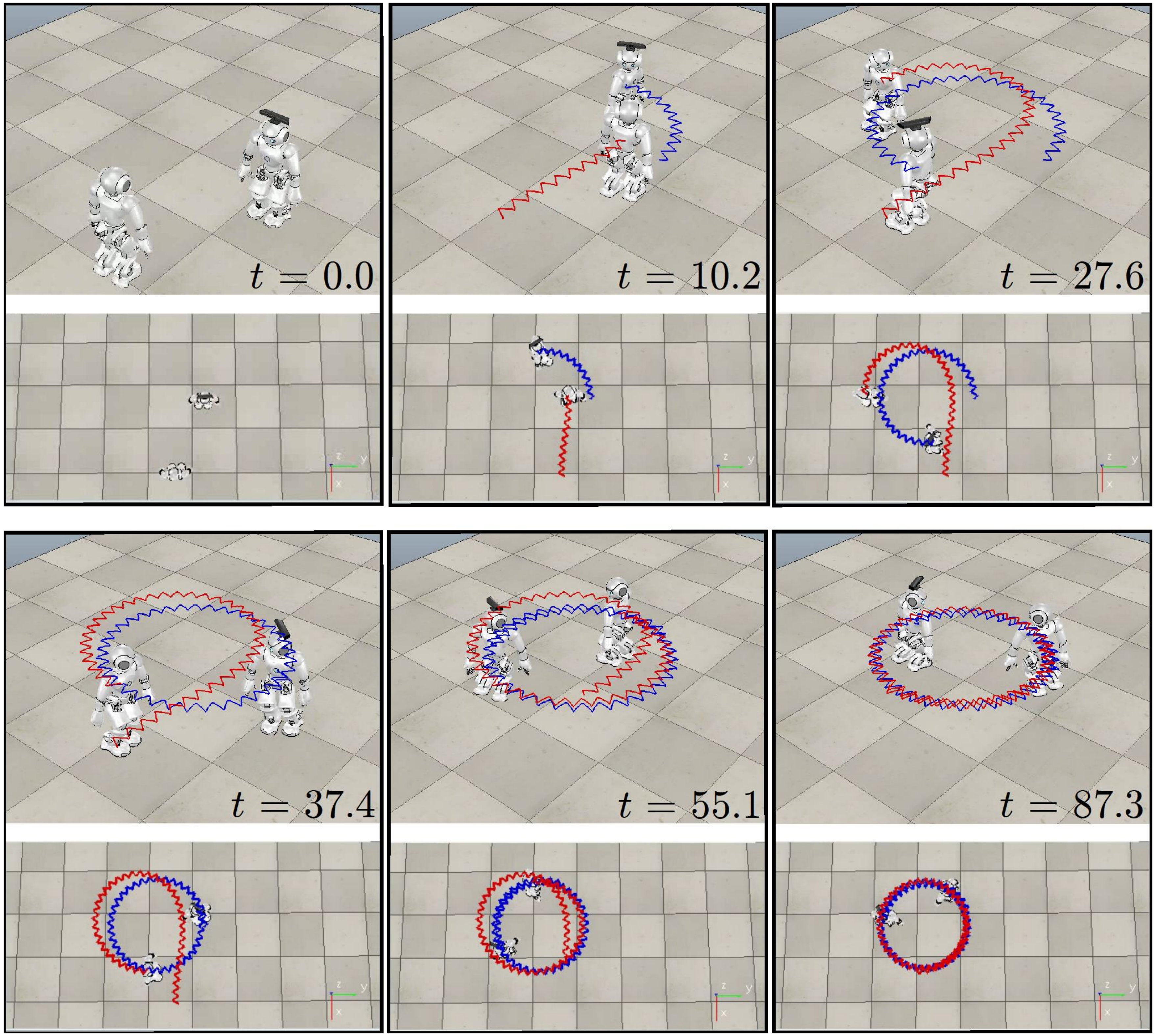
Limit cycle!

Pursuit-Evasion Between Humanoids

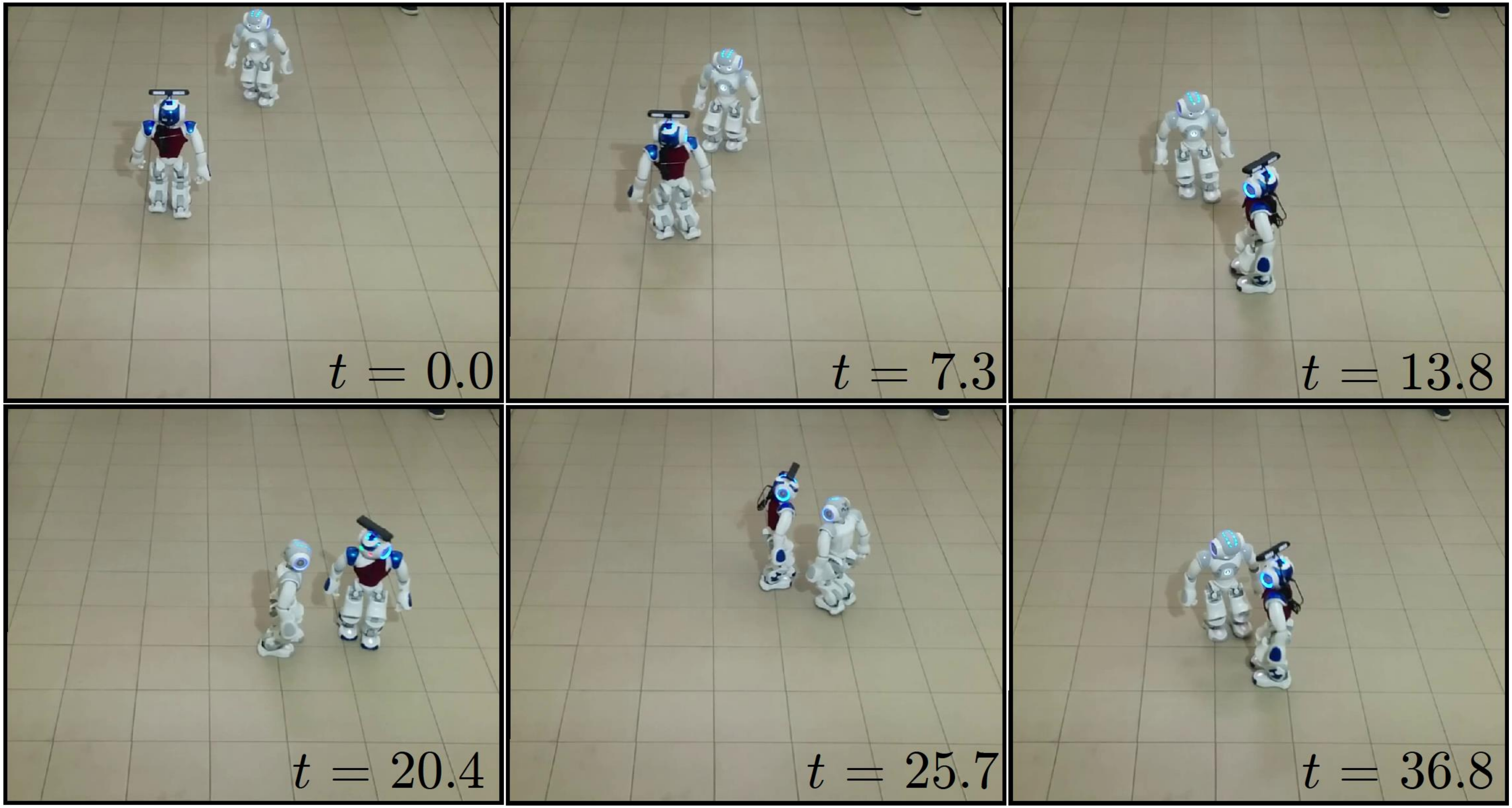
As previously done, the unicycle models are used to generate a reference trajectory for the two humanoids



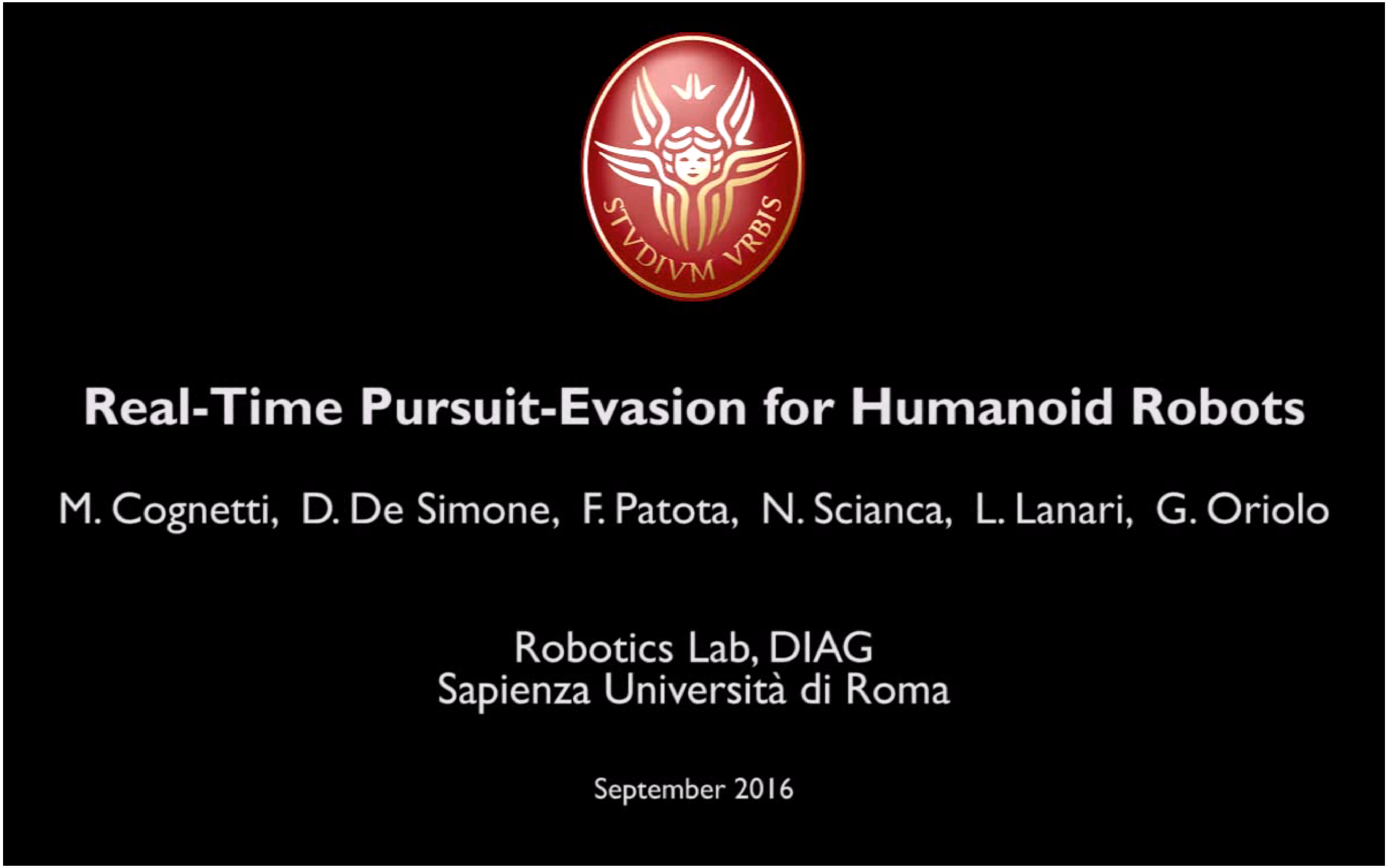
Pursuit-Evasion Between Humanoids - Simulations



Pursuit-Evasion Between Humanoids - Experiments



Wrapping Up

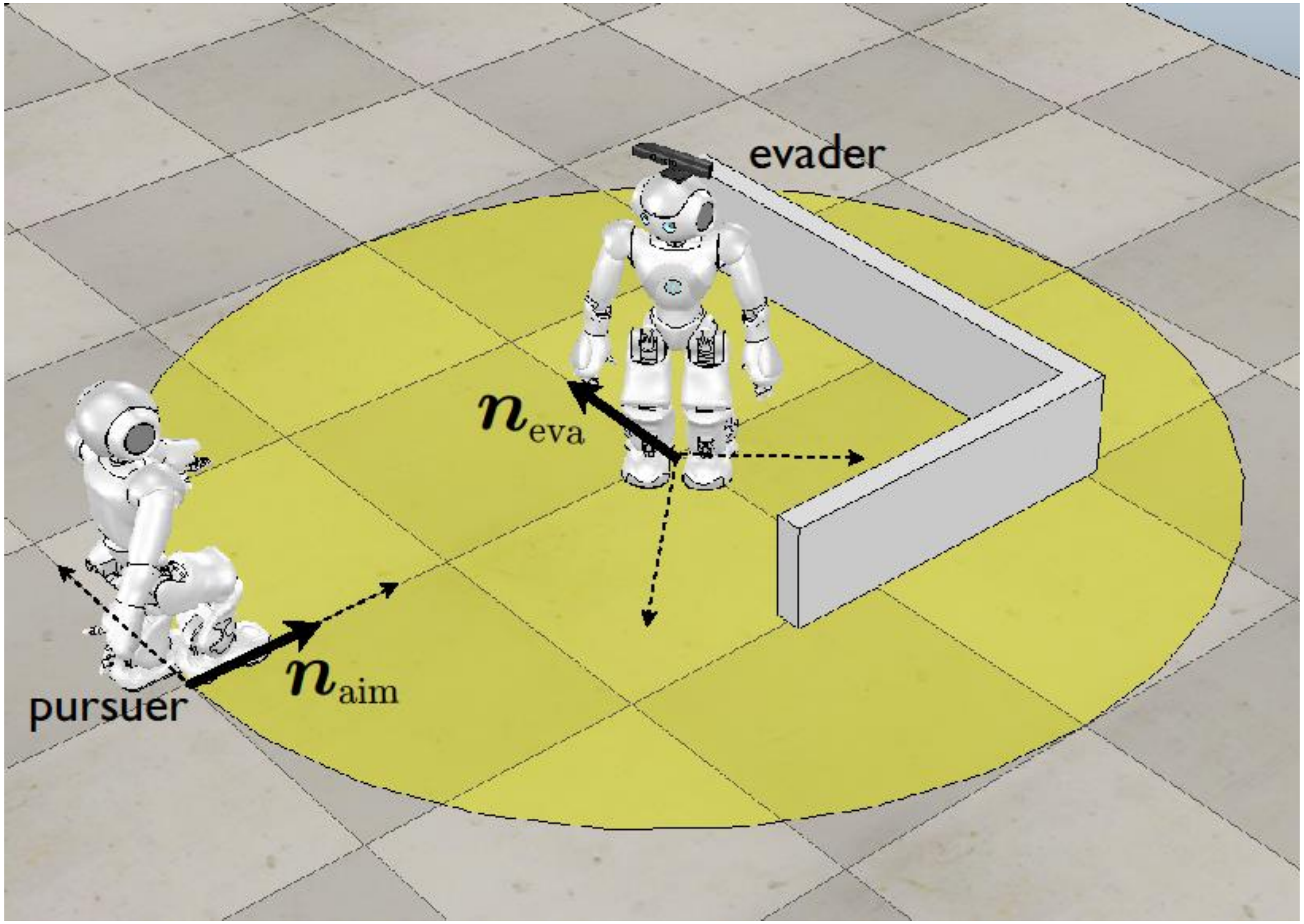


https://youtu.be/nPd2bEPwlIA

Problem Formulation 3

Pursuit-Evasion with obstacles in the environment

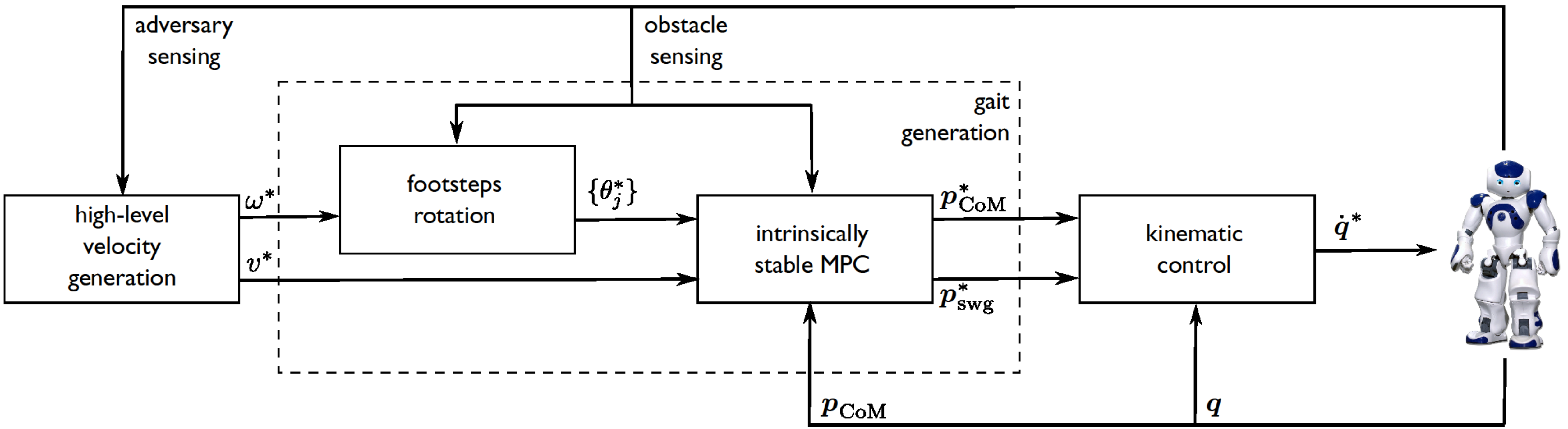
Both robots should plan their movement considering obstacles in **real time**



Model Predictive Control for Gait Generation

Unicycles used as reference model, but this time are used to generate motion commands (i.e. linear and angular velocity)

- Reference velocities are tracked using a MPC scheme
- This allows to take into account obstacles



Model Predictive Control for Gait Generation

Choice of MPC is straightforward because it is more robust and continuously replanned

- Footsteps rotation is chosen to steer away from obstacles
- Introduction of an obstacle constraint as an additional safety layer

Footstep Orientations

Footstep orientation is decided by minimizing

$$\sum_{j=1}^{M}\left(\left(\frac{\theta_j - \theta_{j-1}}{T_s} - \omega\right)^2 + k_{obs}\frac{w(\theta_{obs})}{d^2}\left(\theta_j - \theta_{avo}\right)^2\right)$$

Reference angular velocity tracking

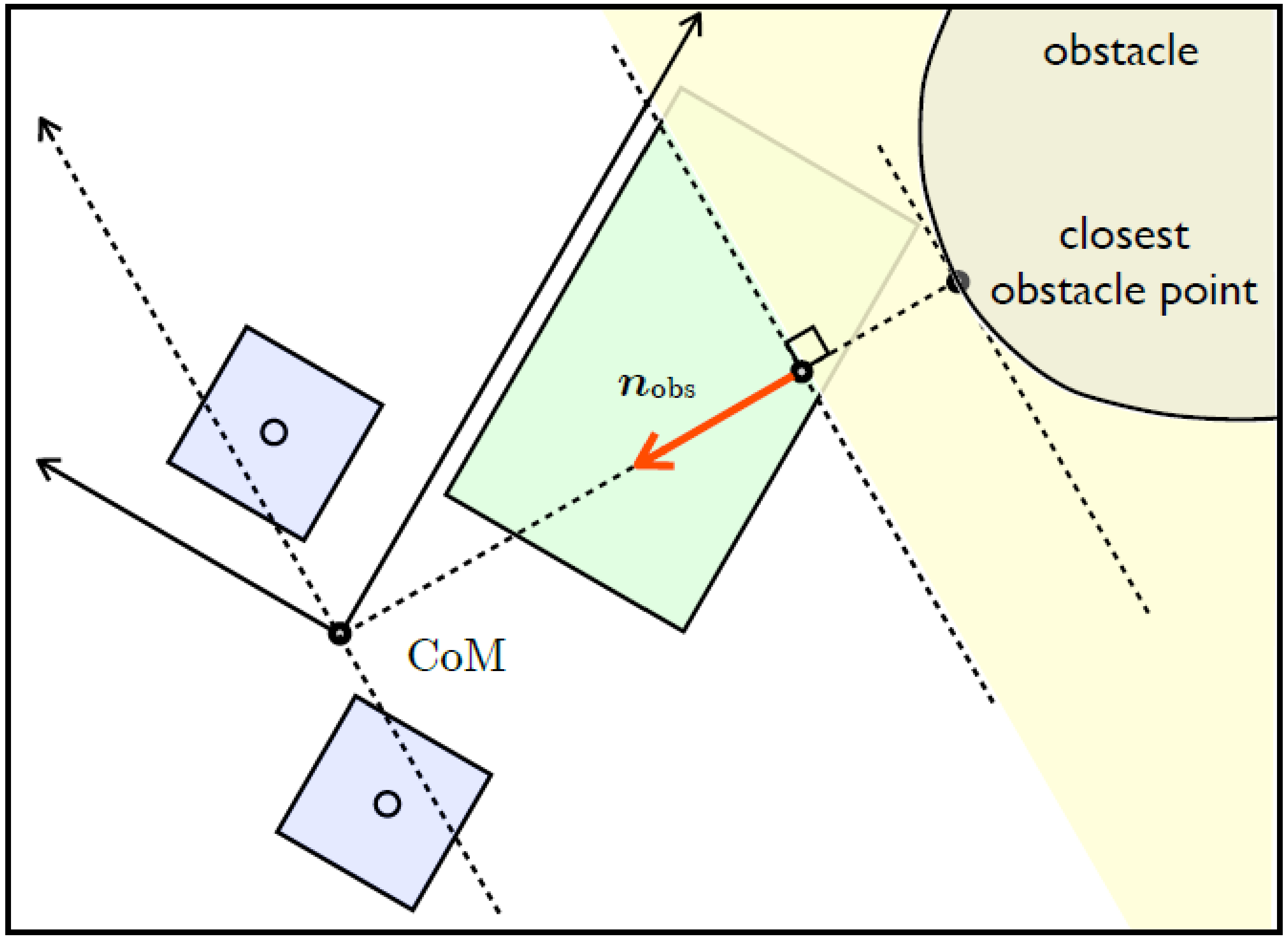
Obstacle avoidance

The obstacle avoidance term fades away when

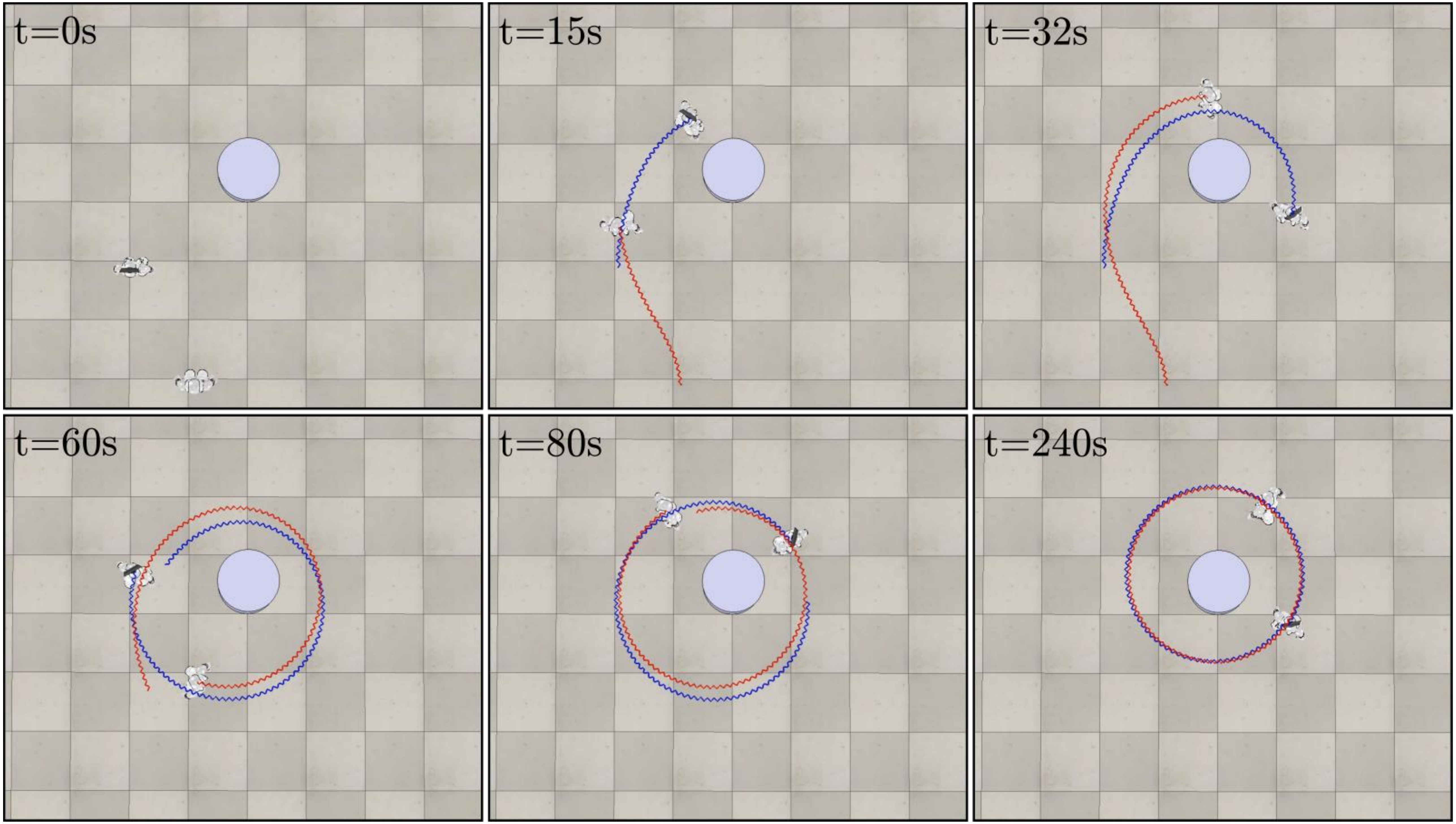
- The obstacle is far away ($1/d^2$)
- The robot is moving away from the obstacle ($w(\theta_{obs})$)

Obstacle Constraint

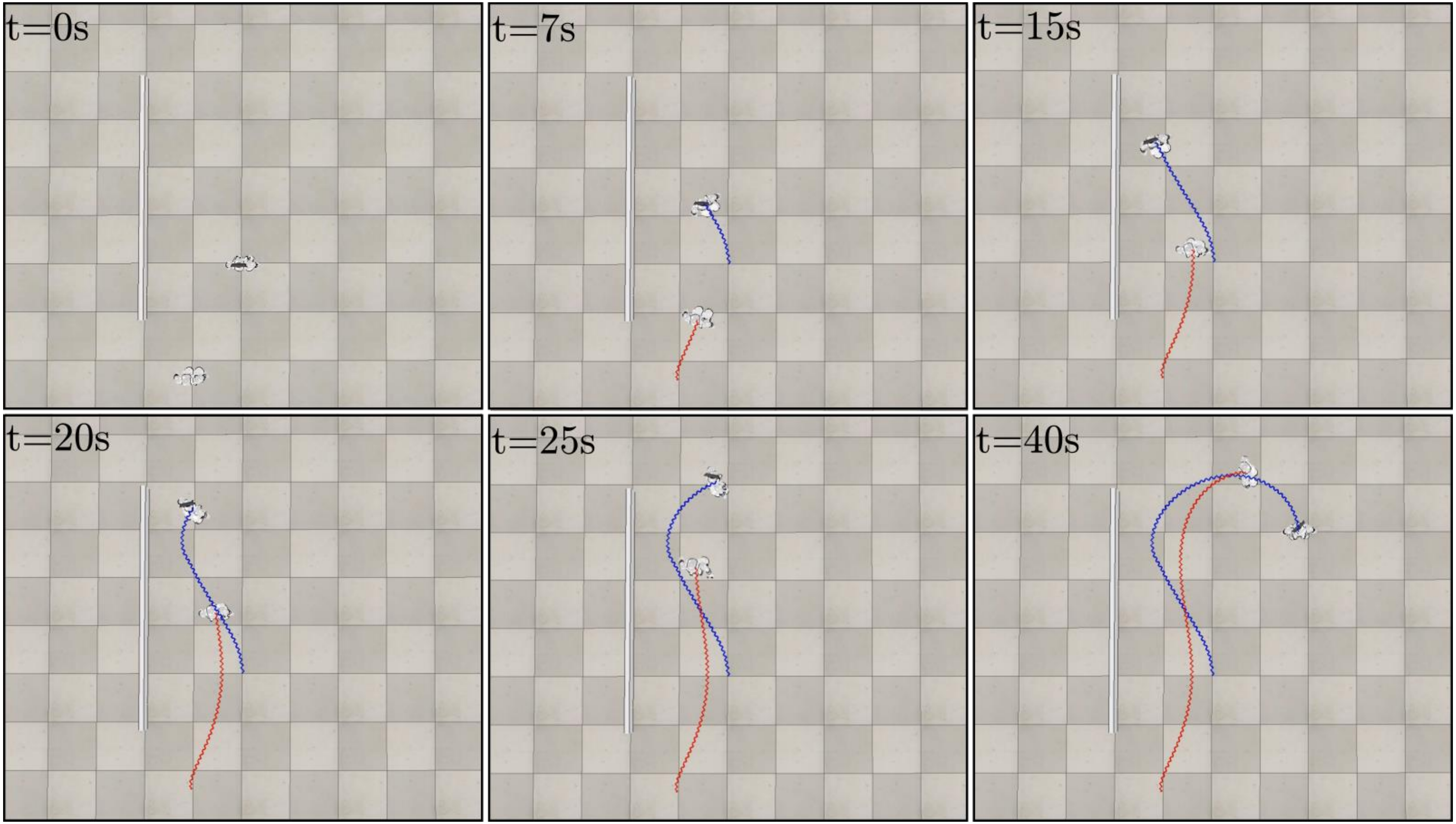
The robot cannot step in the yellow area. The constraint is a half-plane so to preserve linearity. The half-plane is defined by the line perpendicular to n_{obs}



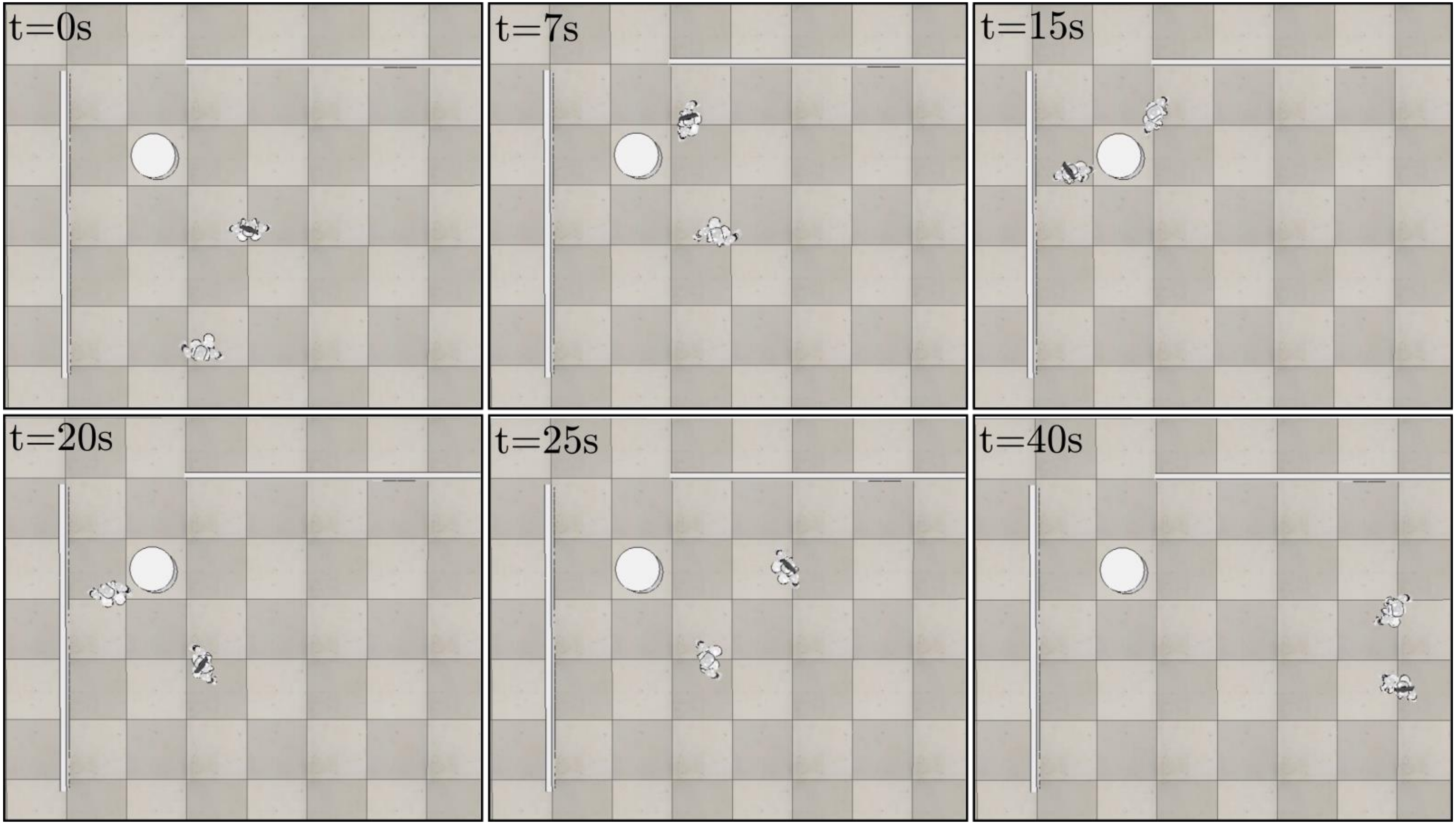
Pursuit-Evasion among Obstacles



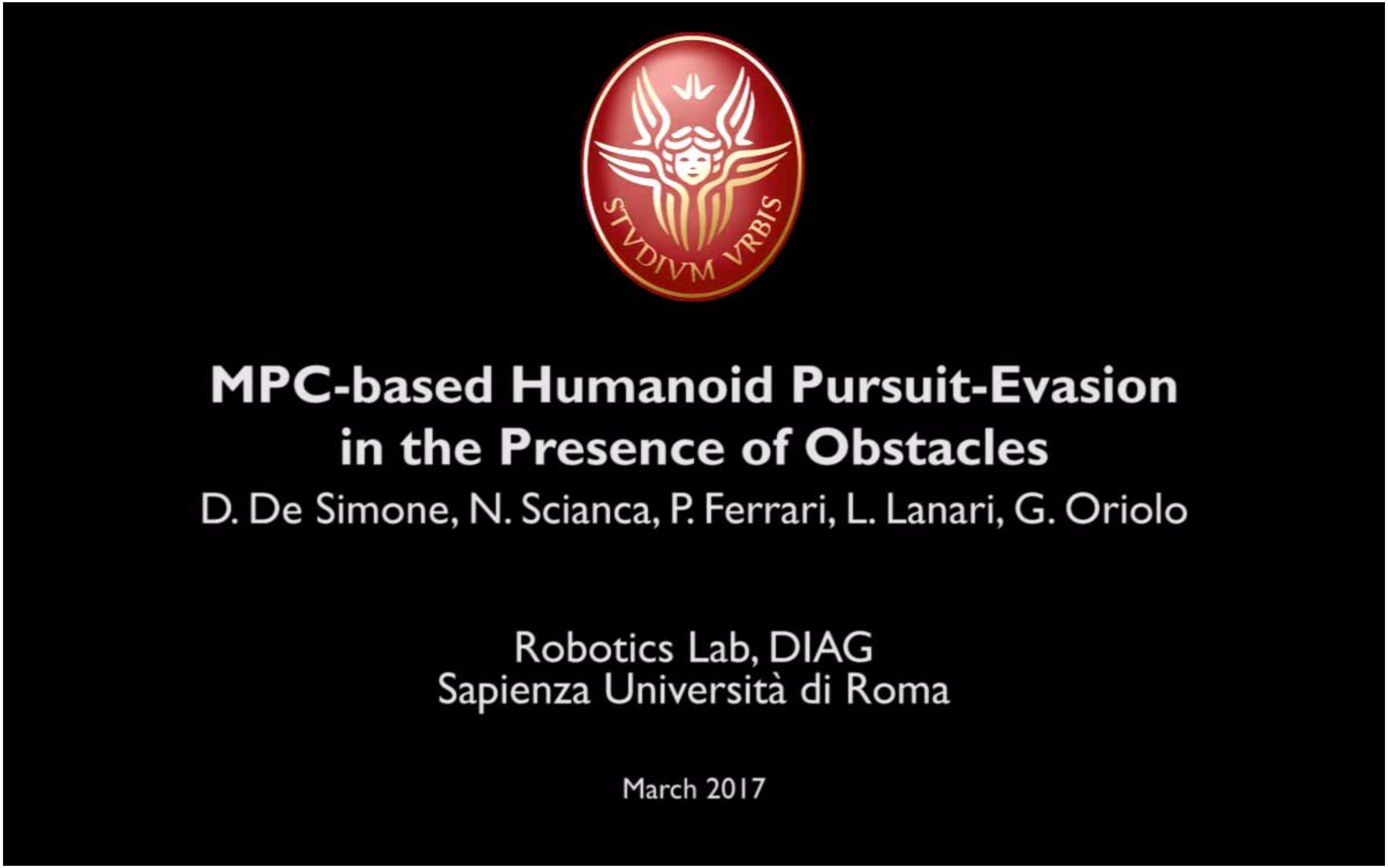
Pursuit-Evasion among Obstacles



Pursuit-Evasion among Obstacles



Wrapping Up



https://youtu.be/oVm6HkofYTc

References

- *Boundedness Issues in Planning of Locomotion Trajectories for Biped Robots* – L. Lanari, S. Hutchinson, L. Marchionni, Humanoids 2014
- *Intrinsically Stable MPC for Humanoid Gait Generation* – N. Scianca, M. Cognetti, D. De Simone, L. Lanari, G. Oriolo, Humanoids 2016
- *Real-Time Planning And Execution of Evasive Motions for a Humanoid Robot* – M. Cognetti, D. De Simone, L. Lanari, G. Oriolo, ICRA 2016
- *Real-Time Pursuit-Evasion with Humanoid Robots* – M. Cognetti, D. De Simone, F. Patota, N. Scianca, L. Lanari, G. Oriolo, ICRA 2017
- *MPC-based Humanoid Pursuit-Evasion in the Presence of Obstacles* – D. De Simone, N. Scianca, P. Ferrari, L. Lanari, G. Oriolo, IROS 2017
- *Gait Generation via Intrinsically Stable MPC for a Multi-Mass Humanoid Model* – N. Scianca, V. Modugno, L. Lanari, G. Oriolo, Humanoids 2017
- *Humanoid Gait Generation for Walk-To Locomotion using Single-Stage MPC* – A. Aboudonia, N. Scianca, D. De Simone, L. Lanari, G. Oriolo, Humanoids 2017